FILTER GRAFTING FOR DEEP NEURAL NETWORKS: REASON, METHOD, AND CULTIVATION

Hao Cheng[1*], **Fanxu Meng**[1*], **Ke Li**[1], **Yuting Gao**[1], **Guangming Lu**[2], **Xing Sun**[1†],**Rongrong Ji**[3]
[1] Tencent Youtu Lab, Shanghai, China
[2] Harbin Institute of Technology, Shenzhen, China
[3] XiaMen University, XiaMen, China.

January 15, 2021

ABSTRACT

Filter is the key component in modern convolutional neural networks (CNNs). However, since CNNs are usually over-parameterized, a pre-trained network always contain some invalid (unimportant) filters. These filters have relatively small l_1 norm and contribute little to the output (**Reason**). While filter pruning removes these invalid filters for efficiency consideration, we tend to reactivate them to improve the representation capability of CNNs. In this paper, we introduce filter grafting (**Method**) to achieve this goal. The activation is processed by grafting external information (weights) into invalid filters. To better perform the grafting, we develop a novel criterion to measure the information of filters and an adaptive weighting strategy to balance the grafted information among networks. After the grafting operation, the network has fewer invalid filters compared with its initial state, enpowering the model with more representation capacity. Meanwhile, since grafting is operated reciprocally on all networks involved, we find that grafting may lose the information of valid filters when improving invalid filters. To gain a universal improvement on both valid and invalid filters, we compensate grafting with distillation (**Cultivation**) to overcome the drawback of grafting . Extensive experiments are performed on the classification and recognition tasks to show the superiority of our method. Code is available at *https://github.com/fxmeng/filter-grafting*.

1 Introduction

For the past decade, Deep Neural Networks (DNNs) have demonstrated their great successes in many research areas, including computer vision [1, 2], speech recognition [3], and language processing [4]. However, recent studies show that DNNs have invalid (unimportant) filters [5]. These filters have little effect on the output accuracy and removing them could accelerate the inference of DNNs without hurting much performance. This discovery inspires many works to study how to identify unimportant filters precisely [6] and how to remove them effectively [7, 8].

However, it is unclear whether abandoning such filters and components directly is the only choice. What if such traditional *invalid* filters can be turned into *valid* and help the final prediction? Similar story happens in the ensemble learning like boosting, where a weak classifier is boosted by combining it with other weak classifiers. In this paper, we investigate the possibility of reactivating invalid filters within the network by bringing outside information. This is achieved by proposing a novel filter grafting scheme, as illustrated in Figure 1. Filter grafting differs from filter pruning in the sense that we reactivate filters by assigning them with new weights from outside, while maintaining the number of layers and filters within each layer as the same. The grafted network has a higher representation capability since more filters in the network are involved to process the information flow.

A key step in filter grafting is choosing proper information source, *i.e.*, where should we graft the information from. In this paper, we thoroughly study this question and claim that we should graft the information from outside (other networks) rather than inside (within the network). We train several networks in parallel. During training at certain epoch,

[1]In the author list, * denotes that authors contribute equally; † denotes corresponding authors.

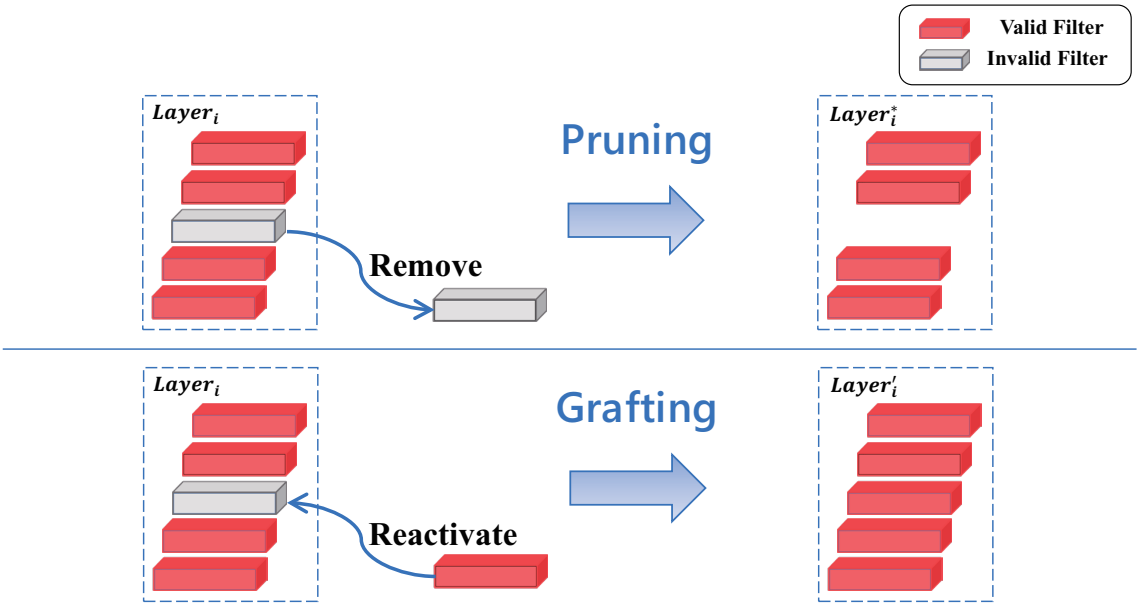

Figure 1: An illustration of the difference between filter pruning and filter grafting. For filter grafting, we graft external information into invalid filters without changing the model structure. (best viewed in color)

Table 1: The difference between filter grafting and other learning methods

methods	without changing model structure ?	one stage ?	without supervision ?
filter pruning [5]	×	×	✓
distillation [10]	✓	×	×
deep mutual learning [11]	✓	✓	×
RePr [12]	✓	×	✓
filter grafting	✓	✓	✓

we graft meaningful filters of a network into invalid filters of another network. By performing grafting, each network could learn external information from other networks, leading to a better representation capability [9]. Meanwhile, since grafting is operated reciprocally on all networks involved, we find that grafting may lose the information of valid filters when improving invalid filters. To overcome this, we experiment grafting with several cultivation ways and surprisingly find that the impact of grafting and distillation are complementary: distillation mostly enhances valid filters while grafting mostly reactivates invalid filters. Thus we compensate grafting with distillation to gain a universal improvement. There are four main contributions of this paper:

- We propose a new learning paradigm called filter grafting for DNNs. Grafting could reactivate invalid filters to improve the potential of DNNs without changing the network structure.
- An entropy based criterion and an adaptive weighting strategy are developed to further improve the performance of filter grafting.
- We analyze the difference between the distillation and the grafting in terms of their effects on network filters, and find that grafting mostly improves the knowledge of invalid filters while distillation mostly improves that of valid filters. This observation inspires us to compensate grafting with distillation to further increase network performance.
- We perform extensive experiments on classification and recognition tasks and show that our method could substantially improve the performance of DNNs.

2 Related Work

Filter Pruning. Filter pruning [13, 14, 15, 16, 17, 18] aims to remove the invalid filters to accelerate the inference of the network. [5] is the first work that utilizes l_1 norm criterion to prune unimportant filters. Since then, more criterions came out to measure the importance of the filters. [19] utilizes spectral clustering to decide which filter needs to be removed. [7] proposes an inherently data-driven method that utilizes Principal Component Analysis (PCA) to specify the proportion of the energy that should be preserved. [20] applies subspace clustering to feature maps to eliminate the redundancy in convolutional filters. Instead of abandoning the invalid filters, filter grafting intends to reactivate them. It is worth noting that even though the motivation of filter grafting is opposite to the pruning, it still involves choosing a proper criterion to decide which filters are unimportant. Thus different criterions used in the pruning are readily applied to the grafting.

Distillation and Mutual Learning. Grafting may involve training multiple networks in parallel. Thus this process is similar to knowledge distillation [10] and mutual learning [11]. Knowledge distillation can be traced back to model compression [21], where authors demonstrate that the knowledge acquired by a large ensemble of models can be transferred to a single small model. Hinton et al. [10] generalize this idea to deep neural networks and show a small, shallow network can be improved through a teacher-student framework. Meanwhile, past works have also been focusing on improving the quality and applicability of knowledge distillation and understanding how knowledge distillation works. The work [22] employs the knowledge transfer learning approach to train RNN using a deep neural network model as the teacher. This is different from the original knowledge distillation, since the teacher used is weaker than the student. Lopes et al. [23] present a method for data-free knowledge distillation, which is able to compress deep neural networks trained on large-scale datasets to a fraction of their size leveraging only some extra metadata with a pre-trained model. The work [24, 25] shows that the student network performance degrades when the gap between student and teacher is large. They introduce multi-step knowledge distillation which employs an intermediate-size network to solve this. The main difference between filter grafting and knowledge distillation is that distillation is a 'two-stage' process. First, we need to train a large model (teacher), then we use the trained model to teach a small model (student). Meanwhile, grafting is a 'one-stage' process, we graft the weight during the training process. However, we demonstrate in the paper that distillation and grafting are complementary on the filter level.

The difference between filter grafting and mutual learning is that mutual learning needs a mutual loss to supervise each network to learn and do not generalize well to multiple networks. While grafting does not need supervised loss and performs much better when we add more networks into the training process. Also, we graft the weight at each epoch instead of each iteration, which greatly reduce the communication cost among networks.

RePr. RePr [12] is similar to our work that considers improving the network on the filter level. However, the motivation of RePr is that there exists unnecessary overlaps in the features captured by network filters. RePr first prunes overlapped filters to train the sub-network, then restores the pruned filters and re-trains the full network. In this sense, RePr is a multi-stage training algorithm. In contrast, the motivation of filter grafting is that the filter whose l_1 norm is smaller contributes less to the network output. Thus the filters that each method operates are different. Also grafting is a one-stage training algorithm which is more efficient.

Grafting. Grafting is firstly adopted as an opposite operation against pruning in decision tree [26], which adds new branches to the existing tree to increase the predictive performance. [27] graft additional nodes onto the hidden layer of trained neural network for domain adaption. NGA [28] is proposed to graft front end network onto a trained network to replace its counterpart, which adapts the model to another domain. While in our approach, filters from different networks are weighted and summed to reactivate invalid filters.

To better illustrate how grafting differs from the above learning types. We draw a table in Table 1. From Table 1, filter grafting is a one stage learning method, without changing network structure and does not need supervised loss.

3 The Proposed Approach

This section arranges as follows: In Section 3.1, we study the information source we need during the grafting process; In Section 3.2, we propose two criterions to calculate the information of filters; In Section 3.3, we discuss how to effectively use the information for grafting; In Section 3.4, we extend grafting method to multiple networks and propose entropy-based grafting algorithm. In section 3.5, we propose grafting+ to overcome the drawbacks of grafting.

3.1 Information Source for Grafting

In the remaining, we call the original invalid filters as 'rootstocks' and call the meaningful filters or information to be grafted as 'scions', which is consistent with botany interpretation for grafting. Filter grafting aims to transfer

information (weights) from scions to rootstocks, thus selecting useful information is essential for grafting. In this paper, we propose three ways to get scions.

3.1.1 Noise as Scions

A simple way is to graft gaussian noise $\mathcal{N}(0, \sigma_t)$ into invalid filters, since Gaussian noise is commonly used for weight initialization of DNNs [29, 30]. Before grafting, the invalid filters have smaller l_1 norm and have little effects for the output. But after grafting, the invalid filters have larger l_1 norm and begin to make more effects on the output.

$$\sigma_t = a^t (0 < a < 1) \tag{1}$$

We also let σ_t decrease over time (see (1)), since too much noise may make the model hard to converge. In Section 4.1, we show that Gaussian noise does not contain useful information thus the improvement is limited.

3.1.2 Internal Filters as Scions

We also experiment adding the internal weights of other filters (whose l_1 norm is bigger) into the invalid filters (whose l_1 norm is smaller) of the same network. Specifically, for each layer, we sort the filters by l_1 norm and set a threshold γ. For filters whose l_1 norm are smaller than γ, we treat these filters as invalid ones. Then we graft the weights of the i-th largest filter into the i-th smallest filter. This procedure is illustrated in Figure 2.

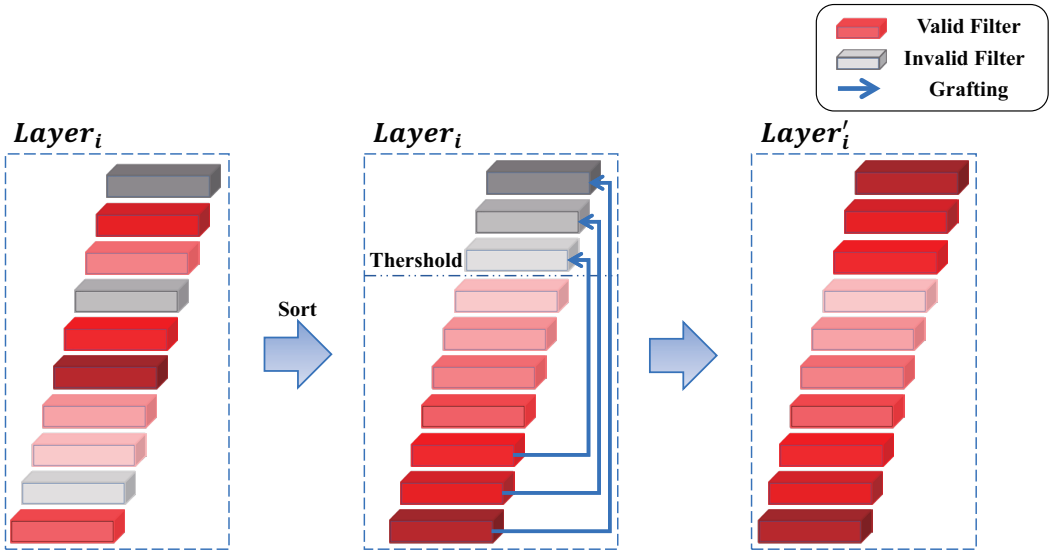

Figure 2: Grafting internal filters. We first sort the filters by l_1 norm, then graft the weights from filters with larger l_1 norm into filters with smaller l_1 norm (best viewed in color).

Since the invalid filters have new weights with larger l_1 norm, they can be activated to have a bigger influence on the output. But this method does not bring new information to the network since the weights are grafted from exactly the same network. We further evaluate it via the language of information theory. To simplify the proving process, we deal with two filters in a certain layer of the network (See Theorem 1). From Theorem 1, selecting internal filters as scions does not bring new information. The experiment in Section 4.1 is also consistent with our analysis.

Theorem 1 *Suppose there are two filters in a certain layer of the network, denoted as random variables X and Y. Z is another variable which satisfies $Z = X + Y$, then $H(X, Y) = H(X, Z) = H(Y, Z)$, where H denotes the entropy from information theory.*

Proof 1 *We first prove* $H(Z|X) = H(Y|X)$*:*

$$\begin{aligned}
&H(Z|X) \\
&= -\sum_x p(X=x) \sum_z p(Z=z|X=x) \log P(Z=z|X=x) \\
&= -\sum_x p(X=x) \sum_z p(Y=z-x|X=x) \log P(Y=z-x|X=x) \\
&= -\sum_x p(X=x) \sum_y p(Y=y|X=x) \log P(Y=y|X=x) \\
&= H(Y|X)
\end{aligned}$$

Then, according to the principle of entropy, we have:

$$\begin{aligned}
H(X,Y) &= H(X) + H(Y|X) \\
&= H(X) + H(Z|X) \\
&= H(X,Z).
\end{aligned}$$

By the symmetry of entropy, the other direction also holds. Thus:

$$H(X,Y) = H(X,Z) = H(Y,Z).$$

3.1.3 External Filters as Scions

In response to the shortcomings of adding random noise and weights inside a single network, we select external filters from other networks as scions. Specifically, we could train two networks, denoted as M_1 and M_2, in parallel. During training at certain epoch, we graft valid filters' weights of M_1 into invalid filters of M_2. Besides, compared to the grafting process in Section 3.1.2, we make two modifications:

- The grafting is processed at layer level instead of filter level, which means we graft the weights of all the filters in a certain layer in M_1 into the same layer in M_2, *vice versa*. Since two networks are initialized with different weights, the location of invalid filters are statistically different and only grafting information into part of filters in a layer may break layer consistency (see more analyses and experimental results in the supplementary material). By performing grafting, the invalid filters of two networks can learn mutual information from each other.
- When performing grafting, the inherent information and the extoic information are weighted. Specifically, We use $W_i^{M_2}$ denotes the weights of the i-th layer of M_2, $W_i^{M_2^{'}}$ denotes the weights of the i-th layer of M_2 after grafting. Then:

$$W_i^{M_2^{'}} = \alpha W_i^{M_2} + (1-\alpha) W_i^{M_1} \quad (0 < \alpha < 1). \tag{2}$$

  Suppose $W_i^{M_2}$ is more informative than $W_i^{M_1}$, then α should be larger than 0.5.

The two-networks grafting procedure is illustrated in Figure 3. There are two key points in grafting: 1) how to calculate the information of $W_i^{M_1}$ and $W_i^{M_2}$; 2) how to decide the weighting coefficient α. We thoroughly study these two problems in Section 3.2 and Section 3.3. Also, we hope to increase the diversity of two networks, thus two networks are initialized differently and some hyper-parameters of two networks are also different from each other (*e.g.*, learning rate, sampling order of data, *etc*). It is worth noting that when performing grafting algorithm on two networks, the two networks have the same weights after grafting process from (2). Grafting is only performed at the end of each epoch, and for other iteration steps, since the two networks are learned with different hyper-parameters, their weights are different from each other. Also, this weighting problem disappears when we add more networks ($N > 2$) in grafting algorithm. Multiple networks grafting can be found in Section 3.4.

3.2 Criterions for Calculating Information of Filters and Layers

In this section, we study two criterions to calculate the information of filters or layers.

3.2.1 L_1 norm

In previous sections, we use l_1 norm to measure the information of filters. Denote $W_{i,j} \in \mathbb{R}^{N_i \times K \times K}$ as the weight of the j-th filter in the i-th convolutional layer, where N_i is the number of filters in i-th layer. Its l_1 norm can be presented

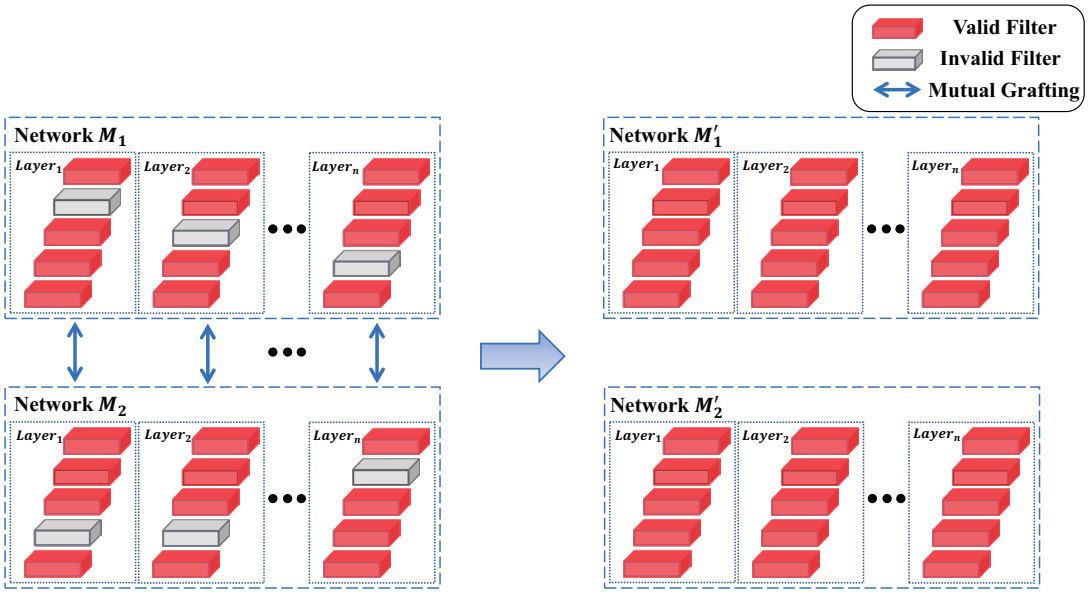

Figure 3: Grafting between two networks. Each network receives the information from the other network (best viewed in color).

Algorithm 1 Entropy-based Multiple Networks Grafting

Input:

Number of networks K, M_k denotes the k-th network; Number of layers L; Training iterations $\mathcal{N} = \{1, \ldots, N_{max}\}$; Number of iterations for each epoch N_T; Training dataset $\mathcal{D}$; Initial weights for each layer of each network $\{\mathbf{W}_l^{M_k} : k = 1, \ldots, K; l = 1, \ldots, L\}$; Different hyper-parameters for each network $\{\lambda_k : k = 1, \ldots K\}$.

Iteration:

for $n = 1$ to N_{max}
  for $\mathbf{k} \in \{1, \ldots K\}, \mathbf{l} \in \{1, \ldots L\}$ **parallel do**
    Update model parameters $\mathbf{W}_l^{M_k}$ based on $\mathcal{D}$ with λ_k //Update model weights at each iteration.
    if $n \ mod \ N_T = 0$
      Get the weighting coefficient α from (7) //Graft model weights at each epoch.
      $\mathbf{W}_l^{M_k} = \alpha \mathbf{W}_l^{M_k} + (1 - \alpha)\mathbf{W}_l^{M_{k-1}}$
    end if
  end for
end for

by:

$$\|W_{i,j}\|_1 = \sum_{n=1}^{N_i} \sum_{k_1=1}^{K} \sum_{k_2=1}^{K} |W_{i,j}(n, k_1, k_2)|. \tag{3}$$

The l_1 norm criterion is commonly used in many researches [5, 31, 32].

3.2.2 Entropy

While l_1 norm criterion only concentrates on the absolute value of filter's weight, we pay more attention to the variation of the weight. We suppose each value of $W_{i,j}$ is sampled from a distribution of a random variable X and use the entropy to measure the distribution. Suppose the distribution satisfies $P(X = a) = 1$, then each single value in $W_{i,j}$ is the same and the entropy is 0. While calculating the entropy of continuous distribution is hard, we follow the strategy from [33, 34]. We first convert continuous distribution to discrete distribution. Specifically, we divide the range of values into m different bins and calculate the probability of each bin. Finally, the entropy of the variable can be calculated as

follows:

$$H(W_{i,j}) = -\sum_{k=1}^{B} p_k \log p_k, \tag{4}$$

where B is the number of bins and p_k is the probability of bin k. A smaller score of $H(W_{i,j})$ means the filter has less variation (information).

Suppose layer i has C filters, then the total information of the layer i is:

$$H(W_i) = \sum_{j=1}^{C} H_{i,j}. \tag{5}$$

One problem of (5) is that it neglects the correlations among the filters since (5) calculates each filter's information independently. To keep the layer consistency, we directly calculate the entropy of the whole layer's weight $W_i \in \mathbb{R}^{N_i \times N_{i+1} \times K \times K}$ as follows:

$$H(W_i) = -\sum_{k=1}^{B} p_k \log p_k. \tag{6}$$

Different from (4), the values to be binned in (6) are from the weight of the whole layer instead of a single filter. In the supplementary material, we prove layer consistency is essential for grafting algorithm.

3.3 Adaptive Weighting in Grafting

In this part, we propose an adaptive weighting strategy for weighting two models' weight from (2). Denote $W_i^{M_1}$ and $H(W_i^{M_1})$ as the weight and information of layer i in network M_1, respectively. The calculation of $H(W_i^{M_1})$ can be referred to (6). We enumerate two conditions that need to be met for calculating the coefficient α.

- The coefficient α from (2) should be equal to 0.5 if $H(W_i^{M_2}) = H(W_i^{M_1})$ and larger than 0.5 if $H(W_i^{M_2}) > H(W_i^{M_1})$.
- Each network should contain part of self information even though $H(W_i^{M_2}) \gg H(W_i^{M_1})$ or $H(W_i^{M_2}) \ll H(W_i^{M_1})$.

In response to the above requirements, the following adaptive coefficient is designed:

$$\alpha = A * (arctan(c * (H(W_i^{M_2}) - H(W_i^{M_1})))) + 0.5, \tag{7}$$

where A and c in (7) are fixed hyper-parameters, and α is the coefficient of (2). We further depict a picture in Figure 4. We can see this function well satisfies the above conditions.

3.4 Extending Grafting to Multiple Networks

Grafting method can be easily extended to a multi-networks case, as illustrated in Figure 5. At the end of each epoch during training, each network M_k accepts the information from M_{k-1}. After certain training epochs, each network contains information from all the other networks. The weighting coefficient is also calculated adaptively. From Section 4.2, we find that by using grafting to train multiple networks, all networks involved could expect a performance gain. We propose our entropy-based grafting in Algorithm 1. It is worth noting that grafting is performed on multiple networks in parallel, which means when we use $\mathbf{W}_l^{M_{k-1}}$ to update $\mathbf{W}_l^{M_k}$, $\mathbf{W}_l^{M_{k-1}}$ has not been updated by grafting yet.

3.5 Cultivating the Grafted Network (Grafting+)

Grafting aims to improve the quality of invalid filters. However, when performing grafting, the weight of invalid filters may also be grafted to the other networks which degrade the knowledge of valid filters. The experimental results in Section 5 also verify this assumption. To alleviate this problem, we employ distillation in grafting system since we find that the impact of grafting and distillation are complementary: distillation mostly enhances valid filters while grafting mostly reactivates invalid filters (verified in Section 4).

Suppose there are C students network and M teachers network, besides performing grafting among student networks, we also perform distillation among teachers and students. That is, except for the base cross-entropy loss, we also add distilled loss on each student and the total loss can be expressed as:

$$L_{s_c} = L_{C_{s_c}} + L_{KD_{t \to s_c}}, \tag{8}$$

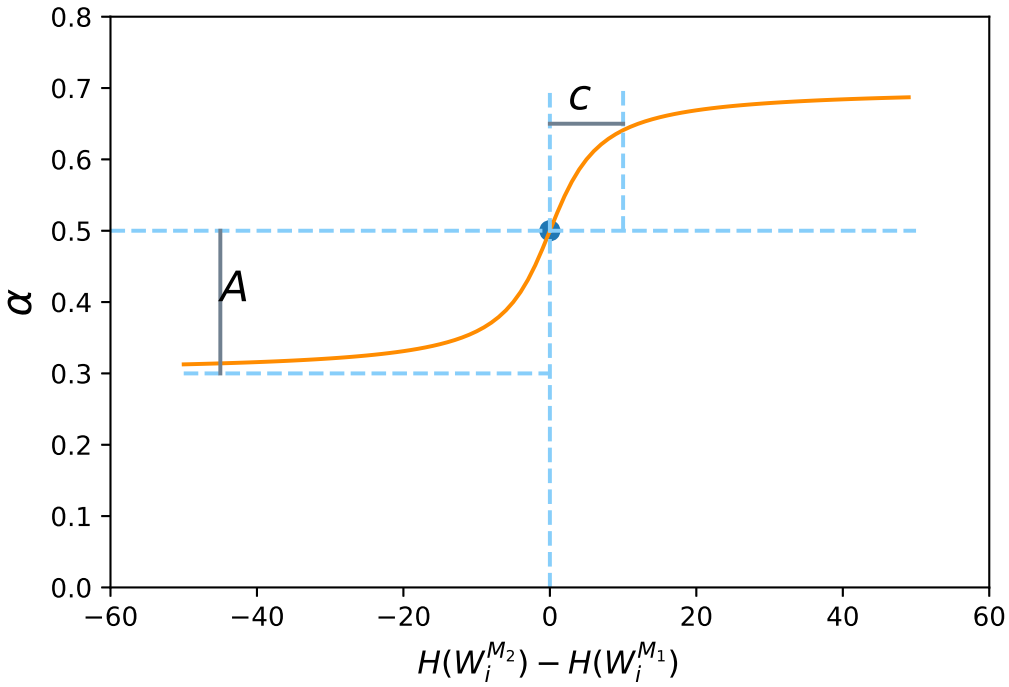

Figure 4: The adaptive coefficient in grafting process.

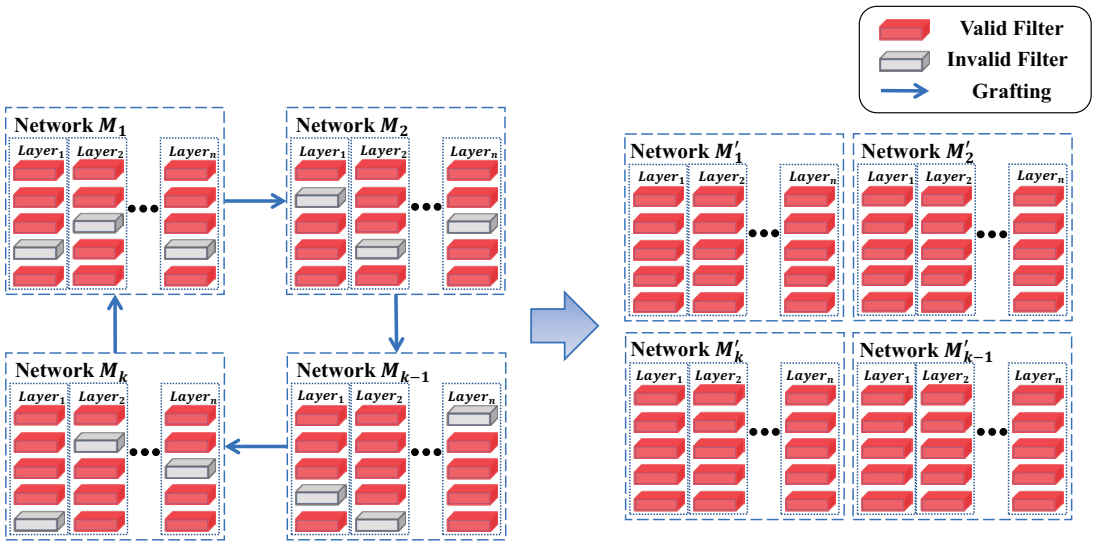

Figure 5: Grafting with multiple networks. The network M_k accepts information from M_{k-1}. (best viewed in color)

where $L_{C_{s_c}}$ is the typical cross entropy loss:

$$L_{C_{s_c}} = -\sum_{i=1}^{N}\sum_{k=1}^{K} I(y_i, k)\log(p_{s_c}^k(x)). \tag{9}$$

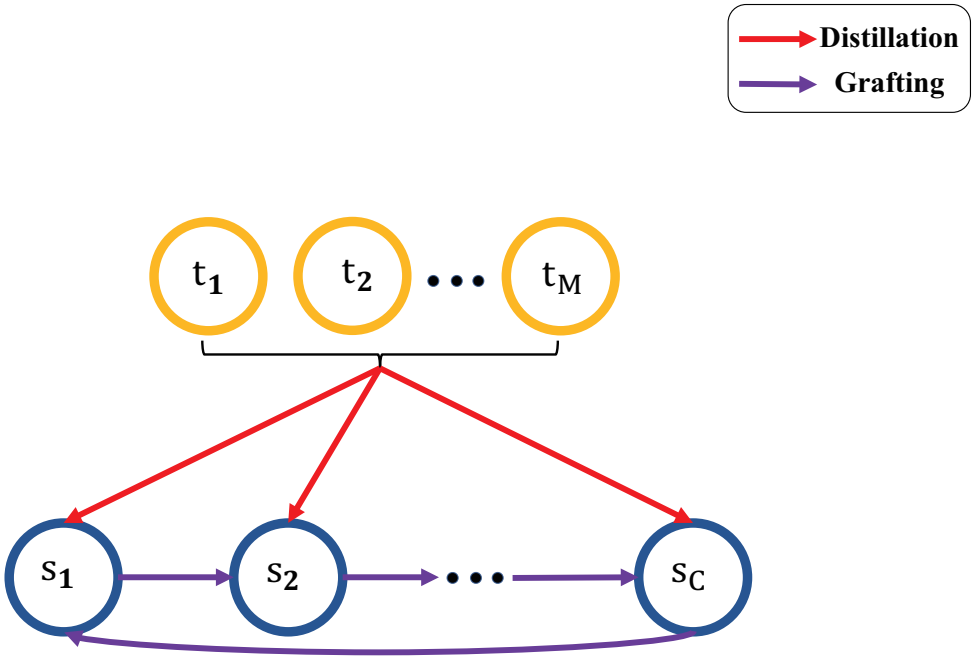

Figure 6: This is the overall framework of grafting+. KD loss is added to the student's cross-entropy loss. (best viewed in color)

The indicator function I is defined as

$$I(y_i, k) = \begin{cases} 1 & y_i = k \\ 0 & y_i \neq k \end{cases} \tag{10}$$

$L_{KD_{t \to s_c}}$ is the knowledge distillation loss defined as:

$$L_{KD_{t \to s_c}} = -\sum_{i=1}^{N} \tau^2 \sum_{k=1}^{K} \bar{p}_k^t(x_i) \log p_k^{s_c}(x_i), \tag{11}$$

where $\bar{p}_k^t(x_i)$ is the average probability of all the teachers given x_i:

$$\bar{p}_k^t(x_i) = \frac{1}{M} \sum_{m=1}^{M} p_k^{t_m}(x_i). \tag{12}$$

The overall framework is presented in Figure 6. In Section 4, we also find that adding more teachers can further increase the performance. It is worth noticing that even though we bring more networks in the framework, the training can be efficiently implemented parallelly. Also we only need one network for inference thus no additional inference cost is required.

4 Experiments

We conduct our experiments in four aspects:

- In Section 4.1, we examine the basic components of grafting, *i.e.*, the information source and the information criterion. Through the experiments, we show that external filters as scions achieves the best result. Also the entropy-based criterion is better than l_1 norm-based criterion.
- In Section 4.2, we perform extensive experiments to verify the effectiveness of grafting and show grafting does improve the invalid filters, leading to a network with better representation ability.

- In Section 4.3, we find that grafting may affect the information of valid filters. However, we observe that the impact of grafting and distillation are complementary: distillation mostly enhances valid filters while grafting mostly reactivates invalid filters. Thus we combines grafting with distillation as a way of cultivation to gain a universal improvement.
- In Section 4.4, we evaluate the performance of grafting together with cultivation and prove that the integral grafting process could universally improve the filters.

4.1 Information Sources and Criterion

Selecting Useful Information Source: We propose three ways to get scions in Section 3 and experimentally examine the three ways on CIFAR-10 and CIFAR-100 datasets [35] in Table 8. Vanilla DNN training without grafting is taken as the baseline. All the methods use MobileNetV2 as the base model. For a fair comparison, the same hyper-parameters are deployed for each method: mini-batch size (256), optimizer (SGD), initial learning rate (0.1), momentum (0.9), weight decay (0.0005), number of epochs (200), learning rate decay (0.1 at every 60 epochs). 'External' here involves training two networks in parallel. In practice, we find the performance of each network in the 'external' method is very close to each other. Thus in the remaining, we always record the first network's performance.

Table 2: Comparison of different scion sources.

	CIFAR-10	CIFAR-100
baseline	92.42	71.44
noise	92.51	72.34
internal	92.68	72.38
external	**92.94**	**72.90**

From Table 2, the performance of 'internal scions' is similar to 'noise', since we prove in Theorem 1 that choosing internal filters as scions does not bring new information to the network. While choosing external filters as scions achieves the best result among the three methods. In the remaining, all the grafting experiments choose external filters as scions.

Comparison of L_1 norm & Entropy Criterions: We propose two criterions to measure the inherent information of filters in Section 3.2. In this part, we quantitatively evaluate the l_1 norm-based grafting and the entropy-based grafting on CIFAR-10 and CIFAR-100 dataset. The results are listed in Table 3. Two networks are used for grafting, with an identical model structure and training hyper-parameters. From Table 3, we can find that, entropy-based grafting beats l_1 norm-based grafting on every model and dataset setting.

Table 3: Comparison of grafting by l_1 norm & entropy.

model	method	CIFAR-10	CIFAR-100
ResNet32	baseline	92.83	69.82
	l_1 norm	93.24	70.69
	entropy	**93.33**	**71.16**
ResNet56	baseline	93.50	71.55
	l_1 norm	94.09	72.73
	entropy	**94.28**	**73.09**
ResNet110	baseline	93.81	73.21
	l_1 norm	94.37	73.65
	entropy	**94.60**	**74.70**
MobileNetV2	baseline	92.42	71.44
	l_1 norm	92.94	72.90
	entropy	**93.53**	**73.26**

In the rest of the experiments, when performing grafting, we always use external filters as scions and apply entropy based criterion.

4.2 Evaluating Grafting on Different Datasets

CIFAR-10 and CIFAR-100: We first compare grafting with other methods on CIFAR-10 and CIFAR-100 dataset. The results are shown in Table 4. For a fair comparison, 'mutual learning' and 'filter grafting' all involve training

Dataset	method	ResNet32	ResNet56	ResNet110	MobileNetV2	WRN28-10
CIFAR-10	baseline	92.83	93.50	93.81	92.42	95.75
	mutual learning [11]	92.80	–	–	–	95.66
	RePr [12]	93.90	–	94.60	–	–
	filter grafting	**93.94**	**94.73**	**94.96**	**94.20**	**96.40**
CIFAR-100	baseline	69.82	71.55	73.21	71.44	80.65
	mutual learning [11]	70.19	–	–	–	80.28
	RePr [12]	69.90	–	73.60	–	–
	filter grafting	**71.28**	**72.83**	**75.27**	**74.15**	**81.62**

Table 4: Comparion of filter grafting with other learning methods. '–' denotes the result is not reported in the corresponding paper.

two networks. The networks in mutual learning and grafting all have the same network structures. The difference between mutual learning and grafting is that mutual learning trains two networks with another strong supervised loss and communication costs are heavy between networks. One should carefully choose the coefficient for mutual supervised loss and main loss when using the mutual learning method. While for grafting, transferring weights does not need supervision. We graft the weights by utilizing entropy to adaptively calculate the weighting coefficient which is more efficient. The results from Table 4 show that filter grafting achieves the best results among all the learning methods. We also examine the effect of multi-networks grafting in Table 5.

Table 5: Grafting with multiple networks (MobileNetV2).

method	CIFAR-10	CIFAR-100
baseline	92.42	71.44
2 models grafting	94.20	74.15
3 models grafting	94.55	76.21
4 models grafting	95.23	77.08
6 models grafting	**95.33**	**78.32**
8 models grafting	95.20	77.76
6 models ensemble	94.09	76.75

As we raise the number of networks, the performance gets better. For example, the performance with 6 models grafting could outperform the baseline by about 7 percent which is a big improvement. The reason is that MobileNetV2 is based on depth separable convolutions, thus the filters may learn insufficient knowledges. Filter grafting could help filters learn complementary knowledges from other networks, which greatly improves the network's potential. Also it is worth noting that the result of 6 models grafting is even better than 6 models ensembles. But unlike ensemble, grafting only maintains one network for testing. However, the performance stagnates when we add the number of models to 8 in grafting algorithm. We assume the cause might be that the network receives too much information from outside which may affect its self-information for learning. How to well explain this phenomenon is an interesting future work.

ImageNet: To test the performance of grafting on a larger dataset, we also validate grafting on ImageNet [36], an image classification dataset with over 14 million images. We compare grafting with the baseline on ResNet18, ResNet34 and ResNet50 models. The baseline hyper-parameters' setting is consistent with official PyTorch setting for ImageNet[1]: minibatch size (256), initial learning rate (0.1), learning rate decay (0.1 at every 30 epochs), momentum (0.9), weight decay (0.0001), number of epochs (90) and optimizer (SGD). To increase the training diversity, we use different learning rates and data loaders for two networks when performing grafting. The other hyper-parameters' setting is consistent with the baseline. The results inTable 6 shows that grafting can also handle larger datasets.

Market1501 and Duke: Grafting is a general training method for convolutional neural networks. Thus grafting can not only apply to the classification task but also other computer vision tasks. In this part, we evaluate the grafting on Person re-identification (ReID) task [37, 38, 39, 40], an open set retrieval problem in distributed multi-camera surveillance, aiming to match people appearing in different non-overlapping camera views. We conduct experiments on two person ReID datasets: Market1501 [41] and DukeMTMC-ReID (Duke) [42, 43]. The baseline hyper-parameters' setting is consistent with [44]: mini-batch size (32), pretrained (True), initial learning rate (0.1), learning rate decay (0.1 at every 20 epochs), number of epochs (60). Besides data loaders and learning rate, the other hyper-parameters' setting is consistent with the baseline. Table 7 shows that for each model and each dataset, grafting performs better

[1] https://github.com/pytorch/examples/tree/master/imagenet

Table 6: Grafting on ImageNet Dataset

model	method	top-1	top-5
ResNet-18	baseline	69.15	88.87
	grafting	**71.19**	**90.01**
ResNet-34	baseline	72.60	90.91
	grafting	**74.58**	**92.05**
ResNet-50	baseline	75.92	92.81
	grafting	**76.76**	**93.34**

than the baseline. Besides, as mentioned before, increasing the number of networks in grafting can further improve the performance.

Table 7: Grafting on ReID Task

model	method	Market1501		Duke	
		mAP	rank1	mAP	rank1
ResNet50	baseline	67.6	86.7	56.2	76.2
	2 models	70.6	87.8	60.8	79.8
	4 models	**73.33**	**89.2**	**62.1**	**79.8**
MobileNetV2	baseline	56.8	81.3	47.6	71.7
	2 models	63.7	85.2	53.4	76.1
	4 models	**64.5**	**85.8**	**54.3**	**76.3**

Effectiveness of Grafting: In this part, we further analyze the effectiveness of the grafting method. To prove grafting does improve the potential of the network, we calculate the number of invalid filters and information gain after the training process. We select MobileNetV2, which is trained on CIFAR-10 with grafting algorithm, for this experiment. The same network structure without grafting is chosen as the baseline. Experimental results are reported in Figure 7 and Figure 8.

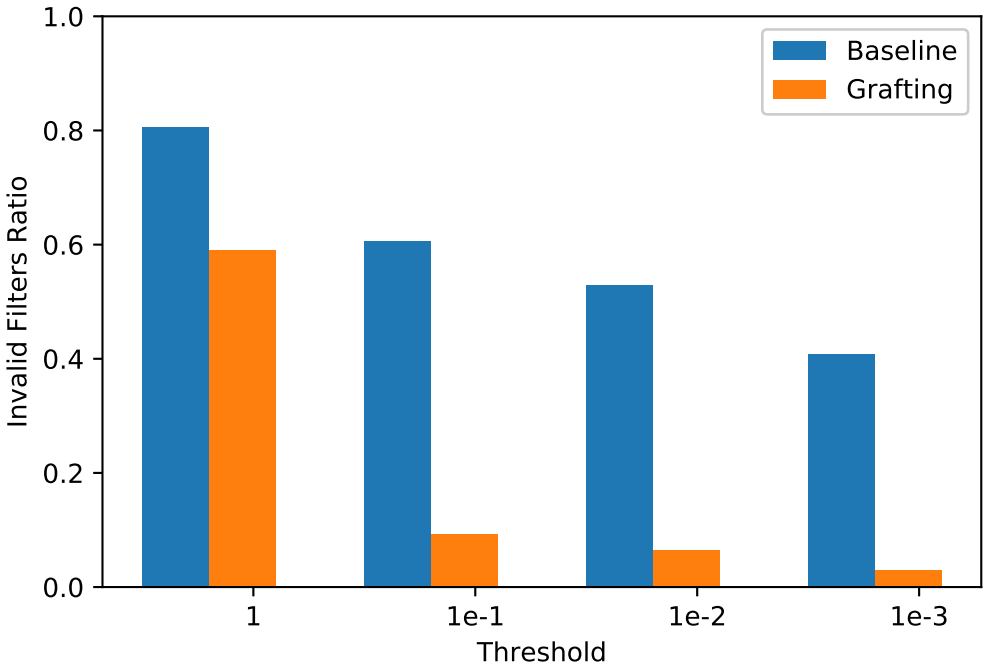

Figure 7: Ratio of filters whose l_1 norm under some threshold.

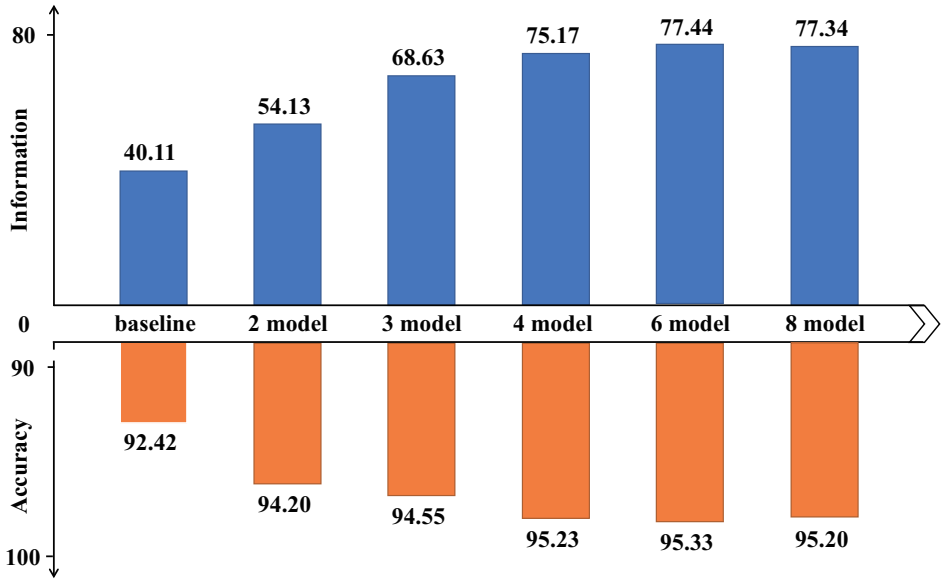

Figure 8: Entropy and accuracy of the baseline network and grafted network. The network's information is defined as the sum of all the layers' entropy in a **single** network. The x axis denotes the number of networks parallelly trained in grafting algorithm.

From Figure 7, under the threshold of 1e-3, there are about 50% filters are invalid or unimportant for the base network, whereas the grafted network only has a small part of filters counted as 'invalid', which shows grafting does help network reduce invalid filters. From Figure 8, the model trained by grafting contains more information than the baseline. Also, the network can gain more information by training multiple networks for grafting method. Thus from the above analysis, we confirm that grafting could improve the potential of neural networks. More analyses can be found in the supplementary material, including the evaluation of invalid filters' locations, necessity of keeping layer consistency and efficiency of adaptive weighting strategy.

Table 8: This table records the density of valid filters and invalid filters for each method.

Dataset	Threshold	Valid filters number	Invalid filters number	Method	Average l_1 norm of valid filters	Average l_1 norm of invalid filters	Test set accuracy
CIFAR-10	0.1	1936	96	baseline	6.982	0.003	93.64
				distillation	**13.648**	0.000	93.70
				grafting	6.427	**1.633**	**94.14**
	1	1920	112	baseline	7.035	0.095	93.64
				distillation	**13.762**	0.000	93.70
				grafting	6.460	**1.757**	**94.14**
CIFAR-100	0.1	1968	64	baseline	12.086	0.001	71.26
				distillation	**26.914**	0.000	71.92
				grafting	10.802	**1.765**	**72.60**
	1	1961	71	baseline	12.126	0.075	71.26
				distillation	**27.011**	0.000	71.92
				grafting	10.830	**1.879**	**72.60**

4.3 Cultivating with Distillation (Grafting+)

From grafting procedure, when we graft multiple networks, even though the information of invalid filters can be replenished by valid filters of other networks, the valid filters may also be affected by invalid filters. It is not clear whether the lost information of valid filters can be recovered during training. Thus in this experiment, we evaluate how distillation and grafting influence the valid and invalid filter. Different thresholds are set to determine which filters are valid or invalid. Specifically, all the filters are ranked according to their l_1 norm values. For filters whose l_1 norms

are larger than the threshold, we consider these filters as valid. Inversely, filters are considered to be invalid if their l_1 norms are smaller than the threshold. Then for each network trained by different methods, we calculate its average l_1 norm of both the valid and the invalid filter. Datasets and training setting are listed below:

Training datasets: The CIFAR-10 and CIFAR-100 datasets are selected for this experiment. The datasets consist of 32 $\times$ 32 color images with objects from 10 and 100 classes respectively. Both are split into a 50000 images train set and a 10000 images test set. The Top-1 accuracy is used as the evaluation metric.

Training setting: The baseline training settings are listed as follows: mini-batch size (256), optimizer (SGD), initial learning rate (0.1), momentum (0.9), weight decay (0.0005), number of epochs (200) and learning rate decay (0.1 at every 60 epochs). Standard data augmentation is applied to the dataset. For knowledge distillation hyper-parameters, we set $\tau = 2$ in (11), which are consistent with the work [45]. For the hyper-parameters regarding filter grafting, we use the same setting from [9], where grafting is performed at the end of each epoch with $A = 0.4$ and $d = 500$ in (7). We use ResNet-56 as the baseline model. Filter grafting involves two ResNet-56 networks that learn knowledge from each other and one network is selected for testing. Knowledge distillation uses ResNet-110 as the teacher and ResNet-56 as the student. After training, the student network is used for testing.

It is worth noting that we identify a filter as valid or invalid by the baseline model with a threshold. To see how grafting and distillation influence filters, the number of valid and invalid filters is fixed for baseline, grafting and distillation. Then we calculate the average l_1 norm of such filters. The results are listed in Table 8. We can see that both knowledge distillation and filter grafting improve the accuracy of the baseline model. However, they behave differently in the filter level. For knowledge distillation, it greatly densifies the knowledge of valid filters and sparsifies invalid filters. For example, the l_1 norms of invalid filters trained by distillation is very close to 0. On the other hand, filter grafting mostly densifies invalid filters since the l_1 norms of invalid filters are prominently improved by the grafting algorithm. We further visualize the filters of the networks trained by each method in Figure 9. We can see that distillation could globally densify the knowledge of the network. However, the network trained by distillation has more invalid filters (l_1 norm close to 0) than baseline. In contrast, grafting could greatly densify the invalid filters and keep more filters functional in networks. This observation means that *the two methods boost the neural network in an opposite way which is naturally complementary.* So it inspires us to design a unified framework to further enhance DNNs. We term the new framework as 'grafting+', a integral grafting process which is shown in Figure 6.

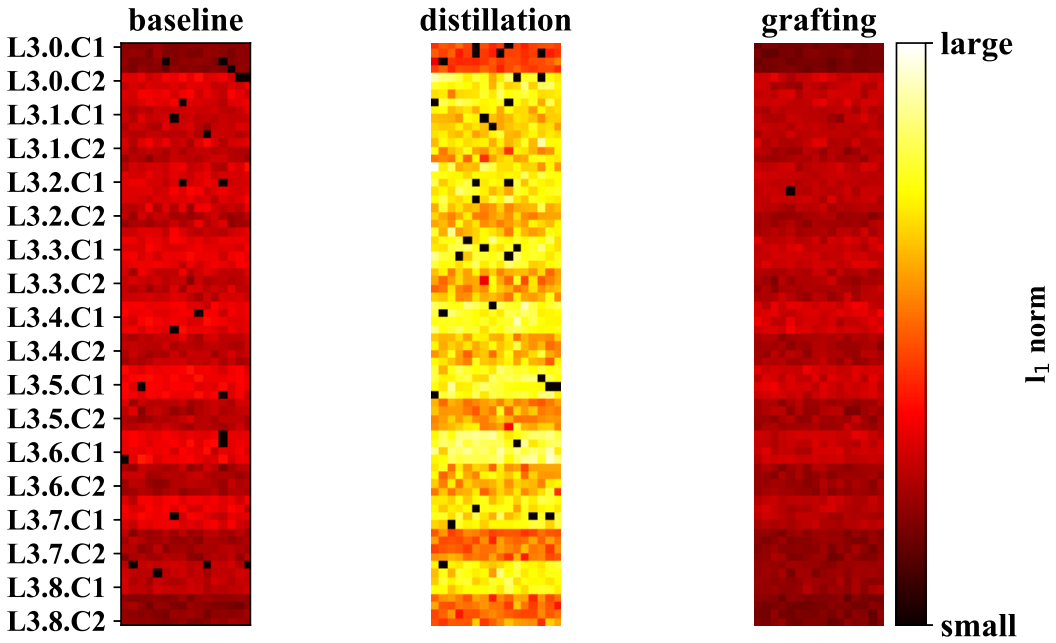

Figure 9: This figure shows the filter weight of each layer. Each filter is represented by a colored cube which is relational to the value of its l_1 norm. The black cubes represent the filters whose l_1 norm are very close to 0. (best viewed in color)

Table 9: This table records the accuracy of CIFAR-100 for each method. Grafting trains two student networks simultaneously while distillation maintains one teacher network and one student network. Grafting+ involves one teacher network and two students networks. For each method, we use the single student network for testing.

Teacher	Student	Method	CIFAR-100
—	ResNet-32	baseline	69.87 ± 0.42
		filter grafting	70.76± 0.33
ResNet-56		knowledge distillation	72.31± 0.23
		GD	**72.69± 0.26**
—	ResNet-56	baseline	71.87± 0.42
		filter grafting	72.00± 0.03
ResNet-110		knowledge distillation	73.42± 0.29
		GD	**74.28± 0.19**
—	MobileNetV2	baseline	72.4± 0.32
		filter grafting	73.79± 0.18
ResNet-110		knowledge distillation	74.98± 0.14
		DGD	**75.55± 0.03**

Table 10: This table records the average l_1 norm of valid filters and invalid filters for each method. The backbone is ResNet-56.

Dataset	Threshold	Method	Average l_1 norm of valid filters	Average l_1 norm of invalid filters	Accuracy on test set
CIFAR-10	0.1	baseline	6.982	0.003	93.64
		grafting+	**12.117**	**1.385**	**94.30**
	1	baseline	7.035	0.095	93.64
		grafting+	**12.191**	**1.635**	**94.30**
CIFAR-100	0.1	baseline	12.086	0.001	71.26
		grafting+	**18.064**	**0.654**	**73.37**
	1	baseline	12.126	0.075	71.26
		grafting+	**18.119**	**0.842**	**73.37**

4.4 Evaluating the Grafting+

grafting+ on CIFAR-100: We compare grafting+ with the baseline, knowledge distillation and filter grafting on CIFAR-100 datasets. The training setting of the baseline, knowledge distillation and filter grafting is introduced in Section 4.3. For grafting+, we use one teacher network and two student networks in this experiment. Knowledge distillation is performed at each optimization step while grafting is performed at the end of each epoch. At the end of the training, we select one student network for testing. For each method on each dataset, we do five runs and report the $mean$ and std of the accuracy. The results are shown in Table 9. We can see that grafting+ gives the best results on CIFAR-100 dataset.

It is worth noting that the setting in this experiment is different with Table 4. In Table 4, the networks in grafting has different learning rate to increase diversity. However, in Table 9, to prove grafting+ can outperform vanilla distillation, the teacher always has better performance than the student, and the learning rate schedule of grafting is consistent with vanilla distillation in this experiment. But we can still observe that the performance of grafting+ on CIFAR-100 is better than grafting in Table 4.

We further depict the accuracy with the training epochs of each method in Figure 10. It's interesting that grafting mostly influences the model at the early training epochs while distillation mostly makes an impact at later stages. Also, the network trained by grafting+ achieves the best accuracy among all the methods.

Extending grafting+ to multiple networks: In previous experiment, grafting+ only considers one teacher network and two student networks. However, this is the simplest form of this framework. We find that grafting+ could further improve the network by bringing more teachers and students to the framework. In this experiment, we use ResNet-110 as the teacher and MobileNetV2 as the student. The model accuracy with the number of teachers and students in grafting+ is listed in Table 11. It can be found that as we raise the number of teachers and students, the model accuracy increases. The results have shown that given more networks, there exists great potential for the grafting+ framework.

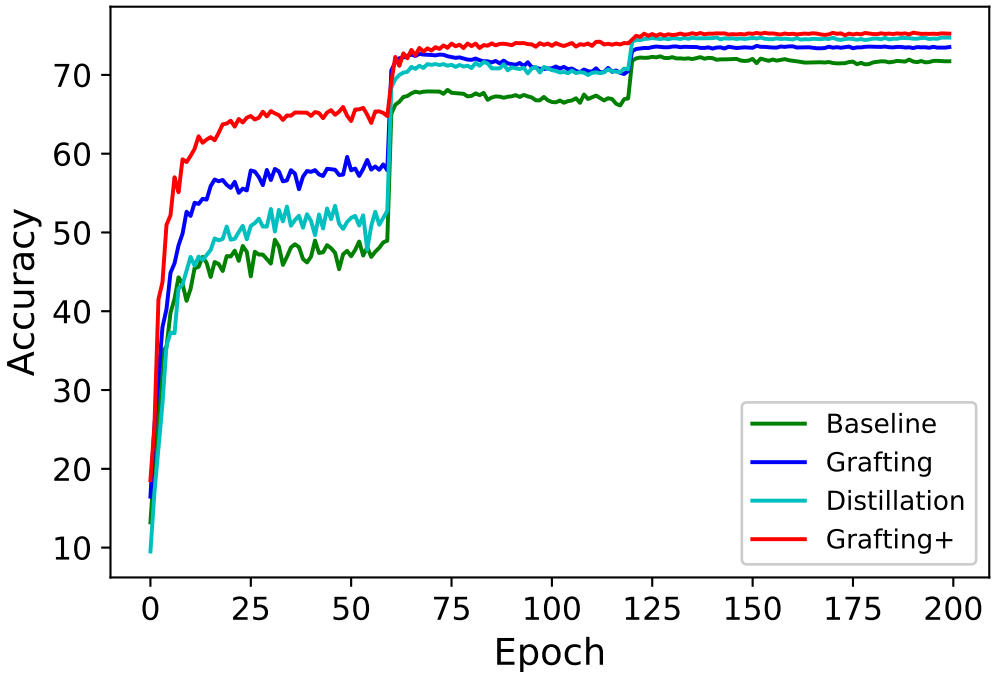

Figure 10: This figure depicts the validation accuracy of each methods on CIFAR-100 datasets. The network is MobileNetV2. (best viewed in color)

Table 11: This table records the student accuracy with number of teachers and students in grafting+ framework. The network structures for teacher and student are ResNet-110 and MobileNetV2, respectively.

Dataset	Setting	Test accuracy
CIFAR-100	teacher*1, student*2	75.18 ± 0.12
	teacher*1, student*4	75.45 ± 0.02
	teacher*1, student*6	76.85 ± 0.02
	teacher*2, student*2	75.96 ± 0.33
	teacher*3, student*2	76.44 ± 0.29

Evaluating grafting+ in the filter level: In order to show that the network trained by grafting+ does improve both valid and invalid filters, we calculate the average l_1 norm of the valid and invalid filters in Table 10 and plot all the filters' l_1 norm in Figure 11. The results show that grafting+ could greatly densify both valid and invalid filters given a fixed model structure. As we state that distillation and grafting are complementary in the filter level, their combination could boost a filter-efficient network.

There are more experiments in the experiments including ablation study on hyper-parameters, more comparison results and visual understanding of grafting+ framework.

5 Conclusion

In this work, a new learning paradigm called 'filter grafting' is proposed. We argue that there are two key points for effectively applying filter grafting algorithm: 1) how to choose proper criterion to calculate the inherent information of filters in DNNs. 2) how to balance the co-efficients of information among networks. To deal with these two problems, we propose entropy-based criterion and adaptive weighting strategy to increase the network's performance. In addition, we find that grafting and distillation has complementary effect on filters. Thus we propose grafting+ framework to improve both valid and invalid filters. Heuristically, there are some future directions to be considered: 1) how to apply

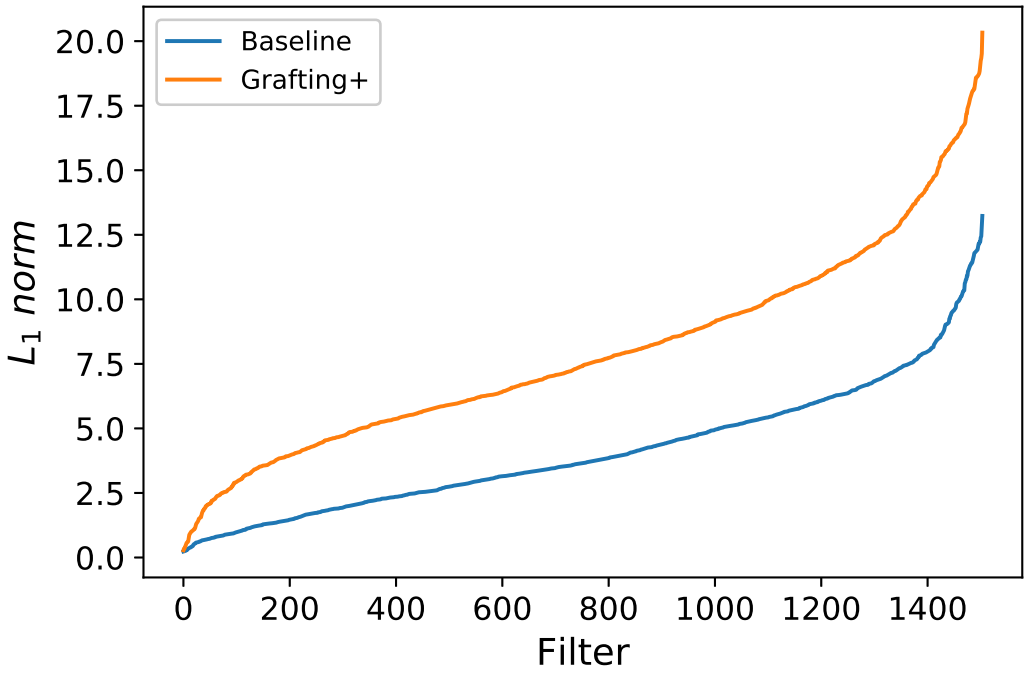

Figure 11: This figure shows the filters' l_1 norm of Conv3 layer for baseline and grafting+. (best viewed in color)

grafting on multiple networks with different network structures; 2) how to improve the network's performance with larger number of networks in grafting+ framework.

References

[1] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[2] William Lotter, Gabriel Kreiman, and David Cox. Deep predictive coding networks for video prediction and unsupervised learning. *arXiv preprint arXiv:1605.08104*, 2016.

[3] Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. Speech recognition with deep recurrent neural networks. In *2013 IEEE international conference on acoustics, speech and signal processing*, pages 6645–6649. IEEE, 2013.

[4] Xiang Zhang and Yann LeCun. Text understanding from scratch. *arXiv preprint arXiv:1502.01710*, 2015.

[5] Hao Li, Asim Kadav, Igor Durdanovic, Hanan Samet, and Hans Peter Graf. Pruning filters for efficient convnets. *arXiv preprint arXiv:1608.08710*, 2016.

[6] Pavlo Molchanov, Arun Mallya, Stephen Tyree, Iuri Frosio, and Jan Kautz. Importance estimation for neural network pruning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 11264–11272, 2019.

[7] Xavier Suau, Luca Zappella, Vinay Palakkode, and Nicholas Apostoloff. Principal filter analysis for guided network compression. *arXiv preprint arXiv:1807.10585*, 2018.

[8] Shaohui Lin, Rongrong Ji, Chao Chen, Dacheng Tao, and Jiebo Luo. Holistic cnn compression via low-rank decomposition with knowledge transfer. *IEEE transactions on pattern analysis and machine intelligence*, 41(12):2889–2905, 2018.

[9] Fanxu Meng, Hao Cheng, Ke Li, Zhixin Xu, Rongrong Ji, Xing Sun, and Guangming Lu. Filter grafting for deep neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6599–6607, 2020.

[10] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

[11] Ying Zhang, Tao Xiang, Timothy M Hospedales, and Huchuan Lu. Deep mutual learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4320–4328, 2018.

[12] Aaditya Prakash, James Storer, Dinei Florencio, and Cha Zhang. Repr: Improved training of convolutional filters. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 10666–10675, 2019.

[13] Song Han, Jeff Pool, John Tran, and William Dally. Learning both weights and connections for efficient neural network. In *Advances in neural information processing systems*, pages 1135–1143, 2015.

[14] Song Han, Huizi Mao, and William J Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149*, 2015.

[15] Yiwen Guo, Anbang Yao, and Yurong Chen. Dynamic network surgery for efficient dnns. In *Advances in neural information processing systems*, pages 1379–1387, 2016.

[16] Zhuang Liu, Jianguo Li, Zhiqiang Shen, Gao Huang, Shoumeng Yan, and Changshui Zhang. Learning efficient convolutional networks through network slimming. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2736–2744, 2017.

[17] Vadim Lebedev and Victor Lempitsky. Fast convnets using group-wise brain damage. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2554–2564, 2016.

[18] Wei Wen, Chunpeng Wu, Yandan Wang, Yiran Chen, and Hai Li. Learning structured sparsity in deep neural networks. In *Advances in neural information processing systems*, pages 2074–2082, 2016.

[19] Huiyuan Zhuo, Xuelin Qian, Yanwei Fu, Heng Yang, and Xiangyang Xue. Scsp: Spectral clustering filter pruning with soft self-adaption manners. *arXiv preprint arXiv:1806.05320*, 2018.

[20] Dong Wang, Lei Zhou, Xueni Zhang, Xiao Bai, and Jun Zhou. Exploring linear relationship in feature map subspace for convnets compression. *arXiv preprint arXiv:1803.05729*, 2018.

[21] Cristian Bucilu, Rich Caruana, and Alexandru Niculescu-Mizil. Model compression. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 535–541. ACM, 2006.

[22] Zhiyuan Tang, Dong Wang, and Zhiyong Zhang. Recurrent neural network training with dark knowledge transfer. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5900–5904. IEEE, 2016.

[23] Raphael Gontijo Lopes, Stefano Fenu, and Thad Starner. Data-free knowledge distillation for deep neural networks. *arXiv preprint arXiv:1710.07535*, 2017.

[24] Seyed-Iman Mirzadeh, Mehrdad Farajtabar, Ang Li, and Hassan Ghasemzadeh. Improved knowledge distillation via teacher assistant: Bridging the gap between student and teacher. *arXiv preprint arXiv:1902.03393*, 2019.

[25] Jang Hyun Cho and Bharath Hariharan. On the efficacy of knowledge distillation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4794–4802, 2019.

[26] Geoffrey I Webb. Decision tree grafting. In *IJCAI (2)*, pages 846–851, 1997.

[27] Yang Li, Xiaoming Tao, and Jianhua Lu. Network grafting: Transferring learned features from trained neural networks. In *2012 IEEE International Conference on Computational Intelligence and Cybernetics (CyberneticsCom)*, pages 40–44. IEEE, 2012.

[28] Yuhuang Hu, Tobi Delbruck, and Shih-Chii Liu. Exploiting event cameras by using a network grafting algorithm. *arXiv preprint arXiv:2003.10959*, 2020.

[29] Siddharth Krishna Kumar. On weight initialization in deep neural networks. *arXiv preprint arXiv:1704.08863*, 2017.

[30] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015.

[31] Jianbo Ye, Lu Xin, Lin Zhe, and James Z. Wang. Rethinking the smaller-norm-less-informative assumption in channel pruning of convolution layers. 2018.

[32] He Yang, Guoliang Kang, Xuanyi Dong, Yanwei Fu, and Yang Yi. Soft filter pruning for accelerating deep convolutional neural networks. In *Twenty-Seventh International Joint Conference on Artificial Intelligence IJCAI-18*, 2018.

[33] Ravid Shwartz-Ziv and Naftali Tishby. Opening the black box of deep neural networks via information. *arXiv preprint arXiv:1703.00810*, 2017.

[34] Hao Cheng, Dongze Lian, Shenghua Gao, and Yanlin Geng. Utilizing information bottleneck to evaluate the capability of deep neural networks for image classification. *Entropy*, 21(5):456, 2019.

[35] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. Cifar-10 (canadian institute for advanced research).

[36] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[37] Weijian Deng, Liang Zheng, Qixiang Ye, Guoliang Kang, Yi Yang, and Jianbin Jiao. Image-image domain adaptation with preserved self-similarity and domain-dissimilarity for person re-identification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 994–1003, 2018.

[38] Longhui Wei, Shiliang Zhang, Wen Gao, and Qi Tian. Person transfer gan to bridge domain gap for person re-identification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 79–88, 2018.

[39] Shan Lin, Haoliang Li, Chang-Tsun Li, and Alex Chichung Kot. Multi-task mid-level feature alignment network for unsupervised cross-dataset person re-identification. *arXiv preprint arXiv:1807.01440*, 2018.

[40] Jingya Wang, Xiatian Zhu, Shaogang Gong, and Wei Li. Transferable joint attribute-identity deep learning for unsupervised person re-identification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2275–2284, 2018.

[41] L. Zheng, L. Shen, L. Tian, S. Wang, J. Wang, and Q. Tian. Scalable person re-identification: A benchmark. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 1116–1124, Dec 2015.

[42] Ergys Ristani, Francesco Solera, Roger Zou, Rita Cucchiara, and Carlo Tomasi. Performance measures and a data set for multi-target, multi-camera tracking. In *European Conference on Computer Vision*, 2016.

[43] Zhedong Zheng, Liang Zheng, and Yi Yang. Unlabeled samples generated by gan improve the person re-identification baseline in vitro. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition(CVPR)*, pages 3754–3762, 2017.

[44] Kaiyang Zhou, Yongxin Yang, Andrea Cavallaro, and Tao Xiang. Omni-scale feature learning for person re-identification. *arXiv preprint arXiv:1905.00953*, 2019.

[45] Chenglin Yang, Lingxi Xie, Chi Su, and Alan L Yuille. Snapshot distillation: Teacher-student optimization in one generation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2859–2868, 2019.