Automatic Network Pruning via Hilbert-Schmidt Independence Criterion Lasso under Information Bottleneck Principle

Song Guo[1]†, Lei Zhang[1]†, Xiawu Zheng[1,2]†, Yan Wang[3], Yuchao Li[4],
Fei Chao[1], Chenglin Wu[5], Shengchuan Zhang[1], Rongrong Ji[1,2,6,7]*
[1]Key Laboratory of Multimedia Trusted Perception and Efficient Computing,
Ministry of Education of China, Department of Artificial Intelligence, School of Informatics,
Xiamen University. [2]Peng Cheng Laboratory. [3]Samsara Inc. [4]Alibaba Group.
[5]Deep Wisdom Inc. [6]Institute of Artificial Intelligence, Xiamen University. [7]Fujian Engineering
Research Center of Trusted Artificial Intelligence Analysis and Application, Xiamen University.
{guosong, leizhang}@stu.xmu.edu.cn, {fchao, zsc_2016, rrji}@xmu.edu.cn
zhengxw01@pcl.ac.cn, grapeot@outlook.com, xiamenlyc@gmail.com, alexanderwu@fuzhi.ai

Abstract

Most existing neural network pruning methods handcrafted their importance criteria and structures to prune. This constructs heavy and unintended dependencies on heuristics and expert experience for both the objective and the parameters of the pruning approach. In this paper, we try to solve this problem by introducing a principled and unified framework based on Information Bottleneck (IB) theory, which further guides us to an automatic pruning approach. Specifically, we first formulate the channel pruning problem from an IB perspective, and then implement the IB principle by solving a Hilbert-Schmidt Independence Criterion (HSIC) Lasso problem under certain conditions. Based on the theoretical guidance, we then provide an automatic pruning scheme by searching for global penalty coefficients. Verified by extensive experiments, our method yields state-of-the-art performance on various benchmark networks and datasets. For example, with VGG-16, we achieve a 60%*-FLOPs reduction by removing* 76% *of the parameters, with an improvement of* 0.40% *in top-1 accuracy on CIFAR-10. With ResNet-50, we achieve a* 56%*-FLOPs reduction by removing* 50% *of the parameters, with a small loss of* 0.08% *in the top-1 accuracy on ImageNet. The code is available at https://github.com/sunggo/APIB.*

1. Introduction

Convolutional Neural Networks (CNNs) [27] have gained great success in computer vision applications such as image classification [17, 51], objective detection [13, 44] and segmentation [37, 4]. However, the high and yet still increasing demands on computing power and memory footprint limit their deployment on edge devices, such as mobile phones or wearable devices. Therefore, many model compression technologies are proposed to compress and accelerate networks including network pruning [16], parameter quantization [5] and knowledge distillation [22]. Among these methods, network pruning has been recognized as an effective tool to support model deployment by reducing the redundancy of neural networks. Prevalent pruning methods can be roughly categorized into weight pruning, channel pruning, and N:M sparsity. Weight pruning, also called unstructured pruning, removes individual weight in weight matrix [10, 7, 28, 16], which requires specific hardware or software and has limited application. Channel pruning [20, 23, 36, 63] breaks this limit by pruning the entire channels and filters, which is more versatile on hardware. N:M sparsity optimizes the sparsity of DNNs so that only N weights are non-zero for every continuous M weights. [67, 65], which also requires specific hardware capabilities to accelerate the networks. In this paper, we focus on channel pruning because of its high adaptability on devices.

The core of channel pruning, which distinguishes different approaches, consists of two aspects: (1) method of channel selection; (2) design of pruning ratio.

Channel Selection. Typical channel pruning methods propose their criteria to measure the importance of channels or filters, such as Norm-based [39, 30, 18], Gradient-based [35, 55], Rank-based [34], BN-based [36], Activation-based [38, 23] and so on. Some other methods [15, 3, 9] assign designed masks or gates for channels as their importance indicators. However, most of these methods have two problems.

*Corresponding author.
†Equal contribution.

First, they are based on heuristics and lack theoretical guidance. Second, They only focus on the inherent properties of individual channels or feature maps, which cannot reasonably interpret network pruning from a global perspective. In this paper, we introduce a novel view from an Information Bottleneck (IB) perspective to channel selection, which supplies the theoretic support for channel pruning and provides a global view across channels.

IB [53] aims to extract the relevant information that an input random variable X contains about an output random variable Y. Let $\tilde{X}$ denote the compressed representation of X. Then the goal of filter pruning is to find the $\tilde{X}$ that preserves the most "relevant" part of X with respect to Y. And the IB is a metric designed to measure this "relevance," even in a non-linear case. In other words, we propose to find the optimal representation $\tilde{X}$ via minimizing the following IB objective

$$I(X;\tilde{X}) - \beta I(\tilde{X};Y)$$

where β is a positive Lagrange multiplier that trades off the complexity of the compression process $I(X;\tilde{X})$, and the amount of the preserved relevant information, $I(\tilde{X};Y)$. While it is ultimately desired to perform a joint optimization on all the layers, as the first trial in this direction, in this paper we perform this IB optimization on each unpruned layer to make it tractable. Then in our context, for a single layer in our method, X denotes the original input feature maps and Y denotes the output feature maps, $\tilde{X}$ represents the pruned input feature maps.

However, because the computation of IB involves density estimation in high dimensional space, in most practical cases, it is computationally infeasible to compute. In this paper, we use HSIC Bottleneck (HB) to approximate estimate IB. HSIC Bottleneck [40] is an implementation of IB principle by replacing the mutual information terms in the IB objective with HSIC [14]. In contrast to mutual information, HSIC can provide a robust computation and does not require density estimation. Thus HB is usually used as an alternative to IB in previous works [40, 43, 59]. And we then prove the equivalence between the HSIC Bottleneck and HSIC Lasso for the first time, which inspires us to utilize the HSIC Lasso to solve channel pruning in a principled way. HSIC Lasso [61] is a kernel-based nonlinear feature selection algorithm with the main concept of minimum redundancy and maximum relevancy (MRMR). It only involves sampled-based Gram matrices, so the prohibitively expensive density estimation could be avoided. As will be shown below, we could use HSIC Lasso to substitute the IB objective under certain circumstances. With further experiments on the application of HSIC Lasso on filter pruning, we found more unique properties of HSIC Lasso: (1) as the batch size increases, the channels selected by HSIC Lasso are almost unchanged; (2) many existing channel pruning methods have highly similar channel selection results, e.g. L_1 norm, L_2 norm [30], FPGM [18], BN-based [36] and Taylor-FO, Taylor-SO [41]. However, the HSIC Lasso can find better channels than the consensus of existing methods.

Pruning Ratio Design. Most pruning methods assume the pruning ratio of each layer a given parameter and thus depend on expert experience to achieve good performance. Our approach solves this problem by treating the pruning ratio for each layer as a prediction output. Specifically, we design a global penalty coefficient λ^* to control the sparsity level, which is shared by all the convolution layers and acts as the coefficient of the HSIC Lasso regularization item. Given a target number of parameters or FLOPs, the target can be achieved by an automatic search for the value of λ^*. The details are illustrated in Sec. 3.3.

We summarize our contributions as follows:

- We interpret channel pruning based on Information Bottleneck theory, which provides theoretic guidance for network pruning, and hints at the limitation of Norm-based approaches. To our best knowledge, this is the first paper to build the relation between input feature maps and output feature maps based on the Information Bottleneck principle.
- For the first time, we prove the equivalence between HB optimization objective and HSIC Lasso, which paved the path for utilizing HSIC Lasso to prune networks based on IB principle.
- We demonstrate that HSIC Lasso-based pruning yields an excellent performance on various benchmarks, with stability and a surprising difference in the chosen channels from other pruning methods.

We conduct extensive experiments on three benchmarks: CIFAR-10, CIFAR-100 [26] and ImageNet [46]. We test various representative networks such as VGG [49], MobileNet-V2 [47], GoogleNet [51] and ResNet [17]. Many Experiments demonstrate that our method has superior performance than other SOTA pruning methods in both model acceleration and compression.

2. Related Work

Channel Pruning. Weight pruning and N:M sparsity require specific devices and are unfriendly to hardware. Channel pruning has no extra requirement and thus becomes a prevalent pruning method to reduce the redundancy of networks. Typical channel pruning methods propose an importance criterion to measure the importance of channels. Li et al. [30] utilize L_1 norm to indicate the importance of channels and consider smaller-norm-less-important. [18] prunes the centered filters in geometric space. [35] calculates the mean gradient of features and considers the feature with a higher mean gradient is more important. Lin et al.

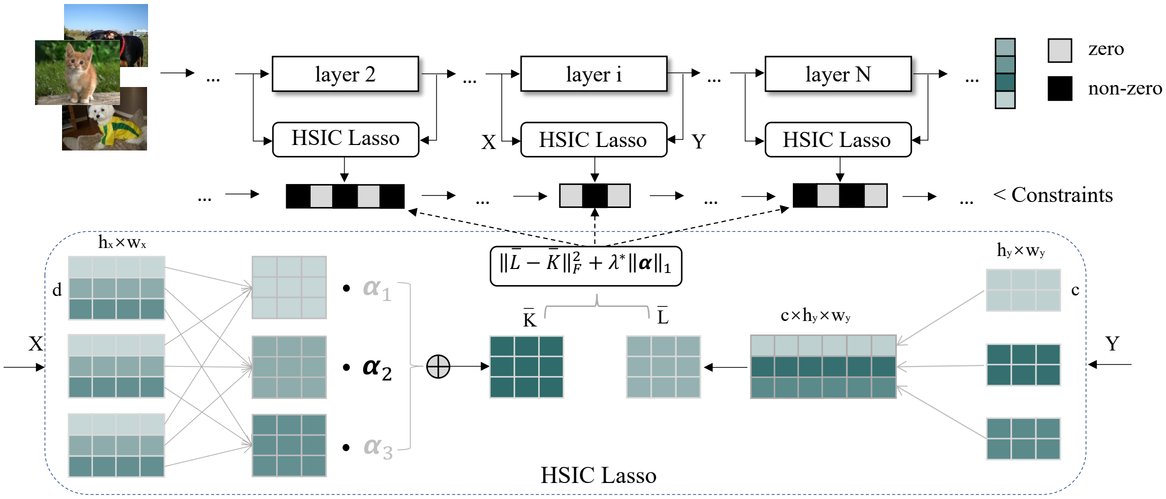

Figure 1. The overview of our proposed method. For each single layer, $\bar{K}$ denotes the sum of the centered Gram matrices for input features X, and $\bar{L}$ denotes the centered Gram matrix for output features Y. Then we optimize the HSIC Lasso loss by Eq. 3. The sparsity coefficient α of unimportant channel tends to be zero.

[34] use the rank of the feature map as the importance criterion, they consider that a low-rank feature map contains less information. However, their method lacks theory basis and ignores the value of information content. [36] uses γ of according BN layer as the importance score of each channel. Hu et al. [23] propose a measure called APOZ to evaluate the importance of each neuron based on the percentage of zero activation. [55] integrates SNIP [29] and Grasp [56] into channel pruning and measures the importance of channels by connection sensitivity and gradient flow. [15, 3, 9] assign a mask or gate for each channel, then update their values during the training stage. On the one hand, most of these methods are heuristic and lack theoretic guidance, they cannot interpret channel pruning from a theoretical and global perspective. On the other hand, these methods manually design pruning ratio, which depends on expert experience and is prone to be trapped in sub-optimal solutions.

Information Bottleneck Principle. The Information Bottleneck (IB) [53] method is an information theoretic principle to extract the output-variables-relevant information in input variables. [54] analyzes the Deep Neural Network (DNN) based on the Information Bottleneck theory. There are some works that apply IB to network pruning. [66] utilize the IB theory to find the pruning ratio of each layer, but they still use heuristic pruning criteria, such as L_1 norm [30] or FPGM [18]. [42] proposed a new objective called NIB to replace origin cross-entropy loss. However, NIB aims to prune individual weight and thus is not friendly to hardware. [48] explores removing filters based on the Mutual Information (MI) between features and labels, [58, 6] use variation Information Bottleneck to compress networks. However, these methods cannot recover their performance after pruning.

3. Method

Notation. Given a pre-trained model M with N convolution layers, let C denote a single convolution layer of M. We change the dimension of the input feature map of C from (n, d, h_x, w_x) to $(n, f = dh_x w_x)$, *i.e.* $X \in \mathbb{R}^{n\times f}$, where n is batch size. Then we convert the shape of output feature maps of C from (n, c, h_y, w_y) to $(n, g = ch_y w_y)$, *i.e.* $Y \in \mathbb{R}^{n\times g}$. Let $p = h_x w_x$, then X is written in the form of block matrix, *i.e.*, $X = [U_1, U_2, \ldots, U_d]$, where $U_k (1 \leq k \leq d) \in \mathbb{R}^{n\times p}$ denotes the k-th channel tensor of X. Therefore, the pruned input features (which is sparse) are represented as $\tilde{X} = [\alpha_1 U_1, \alpha_2 U_2, ..., \alpha_d U_d] \in \mathbb{R}^{n\times f}$, where α_k indicates whether the k-th channel is redundant. If α_k is zero, the k-th channel can be pruned, otherwise, α_k is one indicating that k-th channel can be retained. Our pruning method aims to find and measure the redundant and output-irrelevant channels in X. After pruning, both filters that produce these redundant channels in the previous convolution layer and the input channels that take these redundant channels as input in the next convolution layer both can be removed.

The calculation of Information Bottleneck (IB) can be challenging for several reasons [40, 66]. Firstly, many binning-based algorithms are prone to the curse of dimensionality, which means that different choices of bin size can result in different outcomes. Secondly, introducing a variational distribution [1] to approximate a true distribution can create new sources of noise. To overcome these issues, this paper utilizes HSIC Bottleneck to approximate the IB objective and proves the equivalence between HB optimization and HSIC Lasso. This enables the application of HSIC Lasso to channel pruning based on the IB principle. Moreover, we propose an automatic pruning scheme that searches for the optimal penalty coefficient λ^*. An overview of our method is depicted in Fig. 1.

3.1. Information Bottleneck Objective

For each single layer of M, we formulate our IB objective as:

$$\min -I(\tilde{X};Y) + \gamma I(X;\tilde{X}) \tag{1}$$

where $\tilde{X}$ denotes the pruned input features and γ is a trade-off parameter between the representation complexity $I(X;\tilde{X})$ and the amount of preserved output-relevant information $I(\tilde{X};Y)$. We thus interpret channel pruning as attempts to find the optimal pruned input features that preserve the most "relevant" parts of origin input features with respect to output features.

We then use HSIC Bottleneck [40] to approximate IB by replacing mutual information in Eq. 1 with HSIC. Our IB-based optimation objective can be written as:

$$\min -\text{HSIC}(\tilde{X},Y) + \gamma \text{HSIC}(\tilde{X},X) \tag{2}$$

Similar to IB, HB can remove the redundant information in input X and retain the useful information related to output Y meanwhile. During the pruning process, optimization of Eq. 2 is performed layer by layer via HSIC Lasso to ensure that crucial information is preserved within each layer.

3.2. HSIC Lasso

Hilbert-Schmidt Independence Criterion (HSIC) Lasso [61] is a kernel-based nonlinear feature selection approach that captures input-output nonlinear dependencies. We prove that HSIC Lasso has an equivalent relationship with HB, detailed proof processes are given in Sec. 3.4. The optimization objective of HSIC Lasso is defined as:

$$\min_{\alpha \in R^d} \frac{1}{2}||\bar{\boldsymbol{L}} - \sum_{k=1}^{d} \alpha_k \bar{\boldsymbol{K}}^{(\boldsymbol{k})}||^2_{Frob} + \lambda ||\boldsymbol{\alpha}||_1, \tag{3}$$

$$\text{s.t. } \alpha_1, \dots, \alpha_d \geq 0,$$

where $\bar{\boldsymbol{K}}^{(\boldsymbol{k})} = \boldsymbol{\Gamma}\boldsymbol{K}^{(\boldsymbol{k})}\boldsymbol{\Gamma}$ is the centred Gram matrix for the k-th channel tensor U_k of X, $\boldsymbol{\Gamma} = \boldsymbol{I_n} - \frac{1}{\boldsymbol{n}}\boldsymbol{1_n}\boldsymbol{1_n}^T$; $\boldsymbol{I_n}$ denotes the n-dimensional identity matrix; $\boldsymbol{1_n}$ is denoted as the n-dimensional vector whose elements are all 1; $\boldsymbol{K}^{(\boldsymbol{k})}$ represents the Gram matrix; $\boldsymbol{K}^{(\boldsymbol{k})}_{\boldsymbol{i,j}}$ = K(U_k^i, U_k^j) is the kernel of U_k^i and U_k^j; $U_k^i \in \mathbb{R}^p$ represents the vector of the i-th sample on the k-th channel. Similarly, $\bar{\boldsymbol{L}} = \boldsymbol{\Gamma}\boldsymbol{L}\boldsymbol{\Gamma}$ is the centered Gram matrix of output feature Y; $\boldsymbol{L}$ is also a Gram matrix; $\boldsymbol{L_{i,j}}$ = K(Y^i, Y^j) is the kernel of Y^i and Y^j, and $Y^i \in \mathbb{R}^g$ denotes the output feature of the i-th sample; $\boldsymbol{\alpha}$ is the non-negative sparsity coefficient, and λ is the coefficient of the regularization item to control the sparsity level of $\boldsymbol{\alpha}$.

Algorithm 1: HSIC Lasso

input : Channel threshold Ω; Penalty coefficient λ;
Pretrained model M with N convolution layers;
output: Optimized pruned model M^*
Initial $M^* \leftarrow$ Copy of M;
for *layer* $q = 2$ **to** N **do**
    $d_q \leftarrow$ Number of input channels in q-th layer;
    if $d_q \geqslant \Omega$ **then**
        $\{\alpha_i\}_{i=1}^{d_q} \leftarrow$ Optimize α by Eq. 3;
        Remove i-th channels in q-th layer of M^* If $\alpha_i = 0$;
    end
end
return M^*;

The first item of Eq. 3 is rewritten as:

$$\frac{1}{2}||\bar{\boldsymbol{L}} - \sum_{k=1}^{d} \alpha_k \bar{\boldsymbol{K}}^{(\boldsymbol{k})}||^2_F = \frac{1}{2}\text{HSIC}(Y,Y) - \sum_{k=1}^{d} \alpha_k \text{HSIC}(U_k,Y) + \sum_{k=1}^{d}\sum_{l=1}^{d} \alpha_k \alpha_l \text{HSIC}(U_k,U_l), \tag{4}$$

where HSIC$(U_k, Y) = tr\left(\bar{\boldsymbol{K}}^{(\boldsymbol{k})}\bar{\boldsymbol{L}}\right)$ is a kernel-based independence measure; $tr(\cdot)$ is the trace of matrix. If the k-th channel tensor U_k has a strong dependence on output feature Y, HSIC(U_k, Y) has a high value. Accordingly, α_k takes a large value. If U_k is independent on Y, HSIC(U_k, Y) and α_k tend to zero. Thus, the output-irrelevant channels can be eliminated by HSIC Lasso. Moreover, if U_k and U_l have strong dependences on each other, HSIC(U_k, U_l) is high and thus either of α_k and α_l tends to zero, which tends to remove the redundant channels. Our proposed HSIC Lasso pruning algorithm is summarized in Alg. 1.

3.3. Automatic Pruning

In this section, we illustrate how our approach utilizes HSIC Lasso to prune networks automatically. We propose to set a hyper-parameter λ^* as the global penalty coefficient. λ^* is shared by all convolution layers in a CNN as the coefficient of the HSIC Lasso regularization item, so as to conduct a fair comparison. By increasing λ^*, more zero-value channels will exist in each layer, and the size of the model tends to shrink accordingly. In contrast, if we decrease λ^*, more preserved channels will appear in each layer, and the FLOPs and parameters of pruned model tend to increase. We thus can regard the model size as a monotonically decreasing function of λ^*. The model size can be adjusted by searching the value of λ^*.

If given target FLOPs or parameters, the upper bound is T_u and the lower bound is T_l. Meanwhile, we set a channel number threshold, Ω, and define that each pruned layer has

Algorithm 2: APIB

input : Pretrained model M; Channel threshold Ω; compressed FLOPs/Params range $[T_l, T_u]$;
output: Optimized pruned model M^*
Initial $\lambda_l = 0, \lambda_u = 10^{-6}$;
$t \leftarrow$ FLOPs or Params of M;
while $t > T_l$ **do**
  M^* = `HSICLasso(`λ_u, M, Ω`)`;
  $t \leftarrow$ FLOPs or Params of M^*;
  $\lambda_u = \lambda_u * 2$;
end
while $t \notin [T_l, T_u]$ **do**
  $\lambda^* = \frac{\lambda_u + \lambda_l}{2}$;
  M^* = `HSICLasso(`λ^*, M, Ω`)`;
  $t \leftarrow$ FLOPs or Params of M^*;
  if $t < T_l$ **then**
    $\lambda_r = \lambda^*$;
  else if $t > T_u$ **then**
    $\lambda_l = \lambda^*$;
  end
end
return M^*;

at least Ω preserved channels. If the number of preserved channels is lower than Ω, we stop pruning in this layer; so that Ω prevents the network capacity from damage. If the model size (FLOPs or parameters) of the current pruned model is larger than T_u, we decrease the model size by increasing λ^*. If the current pruned model size is smaller than T_l, we increase the network size by decreasing λ^*. We repeat the above two steps until the current pruned model size satisfies our expectations. In practice, we use a simple binary search algorithm to quickly search the appropriate λ^*. For simplicity, we call our approach APIB. Our proposed automatic pruning method is summarized in Alg. 2.

3.4. Theoretical Analysis

In this section, we first provide proof of the equivalent relationship between HB objective and HSIC Lasso, so as to demonstrate that we can use HSIC Lasso to prune networks based on the IB principle. Then, we further provide new corollaries to prove that Norm-based pruning actually only optimizes the first item of Eq. 2, leading to a sub-optimal solution.

HB and HSIC Lasso. Note that the input features can be written as a block matrix, *i.e.*, $X = [U_1, U_2, \ldots, U_d]$. Eq. 2 is then expanded as:

$$\begin{aligned} & -\text{HSIC}(\tilde{X}, Y) + \gamma \text{HSIC}(\tilde{X}, X) \\ & = -\sum_{k=1}^{d} \alpha_k^2 tr(U_k U_k^T Y Y^T) + \sum_{k=1}^{d}\sum_{l=1}^{d} \gamma_l \alpha_k^2 tr(U_k U_k^T U_l U_l^T), \end{aligned} \tag{5}$$

where we generalize γ to a vector $i.e.$ γ_l = γ, the value of α is zero or one, and thus Eq. 5 can be rewritten as :

$$-\sum_{k=1}^{d} \alpha_k tr(U_k U_k^T Y Y^T) + \sum_{k=1}^{d}\sum_{l=1}^{d} \gamma_l \alpha_k tr(U_k U_k^T U_l U_l^T),$$

If the hyper-parameter γ_l is chosen to variable $\frac{1}{2}\alpha_l$, then we have:

$$\begin{aligned} & -\sum_{k=1}^{d} \alpha_k tr(U_k U_k^T Y Y^T) + \frac{1}{2}\sum_{k=1}^{d}\sum_{l=1}^{d} \alpha_l \alpha_k tr(U_k U_k^T U_l U_l^T) \\ & = -\sum_{k=1}^{d} \alpha_k \text{HSIC}(U_k, Y) + \frac{1}{2}\sum_{k=1}^{d}\sum_{l=1}^{d} \alpha_l \alpha_k \text{HSIC}(U_k, U_l) \end{aligned} \tag{6}$$

If we ignore the constant item, Eq. 6 is the same as Eq. 4. Therefore, while minimizing Eq. 4, we are also minimizing Eq. 2, then, the proof of the equivalence between HSIC Lasso and HSIC Bottleneck is completed.

HB vs. Other Criteria. We also analyze other pruning criteria from the standpoint of information theory. Norm-based pruning is the most prevalent pruning method including L_1 norm, L_2 norm, FPGM, and so on. Norm-based pruning methods tend to prune almost identical filters with the smallest norm [24]. Although Norm-based pruning is widely used to measure the importance of filters, our analysis reveals the limitation of the Norm-based criteria.

Corollary 1 *Norm-based Pruning algorithm is equivalent to maximizing the HSIC*$(\tilde{X}, Y)$ *between the pruned input features* $\tilde{X}$ *and output features* Y.

Proof: We assume the kernel of HSIC is a linear one, Then we have:

$$\text{HSIC}\,(X;Y) = tr((\Gamma X)(\Gamma X)^T(\Gamma Y)(\Gamma Y)^T) \tag{7}$$

where Γ = I_n - $\frac{1}{n} 1_n 1_n^T$; I_n denotes the n-dimensional identity matrix; 1_n is denoted as the n-dimensional vector whose elements are all 1. Note that, ΓX is the centered X, and ΓY is the centered Y; thus Eq. 7 is simplified as:

$$\begin{aligned} HSIC(\boldsymbol{X};\boldsymbol{Y}) &= tr(\boldsymbol{X}\boldsymbol{X}^T\boldsymbol{Y}\boldsymbol{Y}^T) \\ &= ||\boldsymbol{Y}^T\boldsymbol{X}||_{Frob}^2 \end{aligned} \tag{8}$$

where $\boldsymbol{X}$ and $\boldsymbol{Y}$ are centralized matrices; $\boldsymbol{X_i}$ = X_i - $\frac{1}{n}\sum_i X_i$ denotes the i-th sample of $\boldsymbol{X}$; $\boldsymbol{Y_i}$ = Y_i - $\frac{1}{n}\sum_i Y_i$ denotes the i-th sample of $\boldsymbol{Y}$; and n denotes the sample size.

Based on Eq. 8, if we maximize the HSIC$(\tilde{\boldsymbol{X}}, \boldsymbol{Y})$ between the pruned input features $\tilde{\boldsymbol{X}}$ and output features $\boldsymbol{Y}$, we are actually maximizing the distance between $\tilde{\boldsymbol{X}}^{\boldsymbol{T}}\boldsymbol{Y}$ and the zero matrix. We assume the current layer is a fully-connected layer, we then have $\boldsymbol{Y} = W^T\boldsymbol{X}$, where W is the weight matrix, $\boldsymbol{X}$ denotes origin input features.

Model	Method	Baseline Top-1 Acc.	Pruned Top-1 Acc.	FLOPs ↓	Params. ↓	△Top-1 Acc.
VGG-16	L1[30]	93.96%	93.40%	34%	64%	−0.56%
	FPGM[18]	93.96%	93.54%	36%	-	−0.42%
	GDP[15]	93.89%	93.99%	31%	-	+0.10%
	EEMC[64]	93.36%	93.63%	56%	-	+0.27%
	APIB (ours)	**93.68%**	**94.08%**	**60%**	**76%**	**+0.40%**
	CPMC[62]	93.68%	93.40%	66%	-	−0.28%
	PGMPF[3]	93.68%	93.60%	66%	-	−0.08%
	APIB (ours)	**93.68%**	**94.00%**	**66%**	**78%**	**+0.32%**
ResNet-56	Hrank[34]	93.26%	93.17%	50%	42%	−0.09%
	DLRFC[21]	93.06%	93.57%	53%	55%	+0.51%
	SRR-GR[60]	93.38%	93.75%	54%	-	+0.37%
	APIB (ours)	**93.26%**	**93.92%**	**54%**	**50%**	**+0.66%**
	FTWT[9]	93.26%	92.63%	66%	-	−0.63%
	FSM[8]	93.26%	92.76%	68%	68%	−0.50%
	APIB (ours)	**93.26%**	**93.29%**	**67%**	**66%**	**+0.03%**
	DECORE[2]	93.26%	90.85%	81%	85%	−2.41%
	APIB (ours)	**93.26%**	**91.53%**	**81%**	**83%**	**−1.73%**
ResNet-110	Hrank[34]	93.50%	93.36%	58%	59%	−0.14%
	DECORE[2]	93.50%	93.50%	61%	64%	−0.00%
	MPF[19]	93.68%	93.38%	63%	-	−0.30%
	EPruner[33]	93.50%	93.62%	65%	76%	+0.12%
	APIB (ours)	**93.50%**	**94.41%**	**63%**	**65%**	**+0.91%**
	DECORE[2]	93.50%	92.71%	77%	80%	−0.79%
	APIB (ours)	**93.50%**	**93.37%**	**77%**	**82%**	**−0.13%**
GoogleNet	Hrank[34]	95.05%	94.53%	55%	55%	−0.52%
	FSM[8]	95.05%	94.72%	63%	56%	−0.33%
	EPruner[33]	95.05%	94.99%	64%	67%	−0.06%
	APIB (ours)	**95.05%**	**95.29%**	**63%**	**77%**	**+0.24%**

Table 1. Pruning results of VGG-16, ResNet-56, ResNet-110 and GoogleNet on CIFAR-10.

$\tilde{\boldsymbol{X}} = \tilde{W}^T \boldsymbol{X}$, $\tilde{W}$ represents the mask of input features. Thus we have:

$$\begin{aligned} maxHSIC(\tilde{\boldsymbol{X}}, \boldsymbol{Y}) &\iff max||\tilde{\boldsymbol{X}}^T \boldsymbol{Y}||_F^2 \\ &\iff max||\boldsymbol{X}^T \tilde{W} W^T \boldsymbol{X}||_F^2 \\ &\iff max||\tilde{W} W^T||_F^2, \end{aligned} \tag{9}$$

where $\boldsymbol{X}$ is orthogonalized. Norm-based pruning aims to maximize the norm of unmasked weights. Thus we prove that Norm-based pruning is equivalent to minimizing the first item of Eq. 2, *i.e.*, maximizing the output-relevant information in pruned input features, but ignores the optimization of the second item of Eq. 2, *i.e.*, fails to minimize the information that pruned input features contain about origin input features. Thus in Sec. 4, we can observe those Norm-based pruning methods cause sub-optimal performances.

4. Experiments and Analysis

In order to demonstrate the efficiency of our proposed method in both the model compression and acceleration, we conduct extensive experiments to validate many representative CNNs, including VGG, MobileNet-V2, ResNet, and GoogleNet, on three benchmarks: CIFAR-10, CIFAR-100, and ImageNet. We also conduct ablation experiments to further explore APIB.

4.1. Experimental Settings

Training Settings. We train all models by using SGD with 0.9 momentum. For CIFAR-10 and CIFAR-100, our initial learning rate is 0.1 and decayed by the cosine annealing schedule. The weight decay coefficient is set to 2×10^{-4} and batch size is set to 256. After pruning, we train VGG, ResNets, and GoogleNet for 300 to 350 epochs. For Imagenet, our initial learning rate is set to 0.01 and also decayed by the cosine annealing schedule. The weight decay coefficient is set to 1×10^{-4} and batch size is set to 128. We train ResNet-50 and MobileNet-V2 for 150 epochs.

Evaluation Metric. We use the reduction ratio of FLOPs (Float Points Operations) and parameters to evaluate the effectiveness of pruning methods in both model acceleration and compression. Then we provide their Top-1 accuracy to measure their performance on specific tasks.

4.2. Experiments on CIFAR

VGG on CIFAR-10. Tab. 1 presents the results of our experiments on CIFAR-10 using VGG-16. The norm-based pruning method, L1 [30], underperforms and fails to recover accuracy after pruning. In contrast, our proposed APIB outperforms state-of-the-art methods with similar pruning ratios, achieving the highest top-1 accuracy gain when reducing 60% FLOPs and 76% parameters. Specifi-

Model	Method	Baseline Top-1 Acc.	Pruned Top-1	FLOPs ↓	Params. ↓	ΔTop-1 Acc.
ResNet-50	DPFPS[45]	76.15%	75.55%	46%	-	−0.60%
	Random[31]	76.15%	75.13%	49%	54%	−1.02%
	CC[32]	76.15%	75.59%	53%	-	−0.66%
	MFP[19]	76.15%	74.86%	54%	-	−1.29%
	SCOP[52]	76.15%	75.26%	55%	-	−0.89%
	NPPM[11]	76.15%	75.96%	56%	-	−0.19%
	LRF-60[25]	76.15%	75.71%	56%	54%	−0.44%
	APIB (ours)	76.15%	**76.07**%	**56**%	**50**%	**−0.08%**
	DECORE[2]	76.15%	72.06%	61%	-	−4.09%
	Hrank[34]	76.15%	71.98%	62%	62%	−4.17%
	APIB(ours)	76.15%	**75.37**%	**62**%	**58**%	**−0.78%**
MobileNet-V2	DMC[12]	71.80%	68.37%	46%	-	−3.43%
	APS[57]	71.80%	68.96%	48%	-	−2.84%
	APIB(ours)	71.80%	**69.51**%	**51**%	**48**%	**−2.29%**

Table 2. Pruning results of ResNet-50 and MobileNet-V2 on ImageNet.

cally, APIB reduces 66% FLOPs and increases top-1 accuracy from 93.68% to 94.00%, demonstrating an improvement of 0.32%. In comparison, other pruning methods all experience varying degrees of top-1 accuracy loss.

ResNet-56 on CIFAR10. Tab. 1 displays the experimental results of applying APIB and several state-of-the-art pruning methods to ResNet-56 on CIFAR-10. Notably, APIB outperforms all other methods in terms of top-1 accuracy, achieving the largest gain when reducing FLOPs by around 54%. Even when removing 67% of FLOPs, APIB still manages to maintain an improvement in top-1 accuracy. Conversely, other methods such as FSM [8] and FTWT [9] suffer a decrease in top-1 accuracy when pruning. Additionally, APIB continues to yield state-of-the-art performance of 91.53% top-1 accuracy when reducing FLOPs by 81% and parameters by 83%.

ResNet-110 on CIFAR-10. We also validate the effectiveness of our method on ResNet-110, which has a deeper structure than ResNet-56, on CIFAR-10. The results are shown in Tab. 1, where APIB achieves a 63% FLOPs reduction and a 0.91% gain in top-1 accuracy, outperforming other state-of-the-art methods with similar FLOPs reduction ratios. Even when 77% FLOPs and 82% parameters are removed, our APIB only suffers a small loss of 0.13% in top-1 accuracy. In contrast, there is a significant drop in top-1 accuracy for DECORE [2].

GoogleNet on CIFAR-10. We also evaluate the performance of APIB on GoogleNet, which has a multi-branch structure. The results, as shown in Tab. 1, demonstrate that APIB outperforms other pruning methods. APIB achieves a top-1 accuracy of 95.30% with 63% FLOPs reduction and 77% parameters reduction, even surpassing the baseline model with an improvement of 0.24% in top-1 accuracy. However, other methods, such as EPruner [33] and FSM [8], fail to recover the accuracy after pruning.

The experimental results on CIFAR-100 can be found in supplementary.

4.3. Experiments on ImageNet

ResNet-50 on ImageNet. As shown in Tab. 2, under around 55% FLOPs reduction, APIB can reach the 76.07% top-1 accuracy and merely suffers a small loss of 0.08% in top-1 accuracy. However, all other SOTA methods, such as DPFPS [45], SCOP [52], MPF [65], CC [32], NPPM [11], LRF [25] and Random Pruning [31], fail to work on. DECORE [2] and Hrank [34], decrease a lot in top-1 accuracy after removing around 62% FLOPs. In contrast, our APIB has an obviously better performance.

MobileNet-V2 on ImageNet. We perform experiments on ImageNet using MobileNet-V2 and the results are presented in Tab. 2. The pruned network obtained using APIB shows better performance compared to DMC [12] and APS [57] with a similar reduction in FLOPs. These results demonstrate that APIB is also effective for compressing lightweight models.

4.4. Ablation Study

Stability of HSIC Lasso. We prove the equivalent relationship between the HB Objective and HSIC Lasso in Sec. 3.4, then we employ HSIC Lasso to prune networks by applying the Information Bottleneck principle. HSIC Lasso prunes filters by sparsifying input features in a structural manner. We further investigate the impact of sample size, *i.e.*, the number of feature maps on the pruning results of HSIC Lasso. We observe that as the sample size increases, those selected filters are almost unchanged, indicating that the dependence relationship between features tends to be stable. we record the pruning results for hidden layers in ResNet-20, ResNet-44, ResNet-56, and ResNet-110. In Fig. 2, if the number of feature maps is large than 256, the selected filters are almost unchanged.

Influence of sample sizes. To better understand the effect of sample sizes on the final accuracy of pruned models, we conduct experiments during the pruning stage. Our test subjects include ResNet-20, ResNet-44, ResNet-56, and

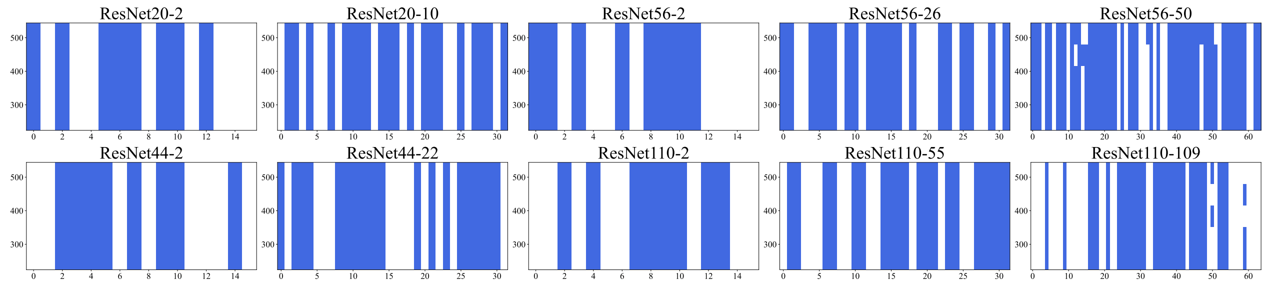

Figure 2. Pruned and preserved filter statics from different convolution layers and networks on CIFAR-10. For each sub-figure, the X-axis represents the indices of filters and the Y-axis represents the number of batch sizes. White denotes preserved filters and blue denotes pruned filters.

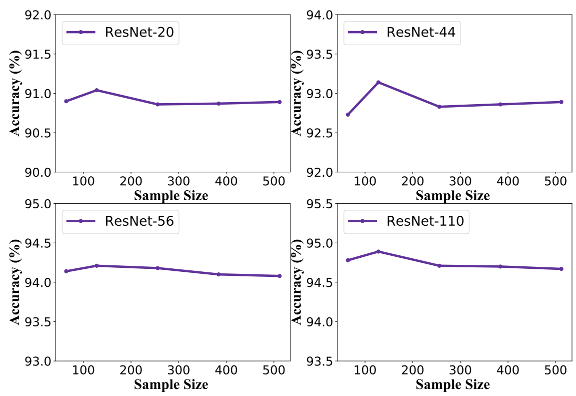

Figure 3. The final accuracy of the pruned model under different sample sizes. We test ResNet-20, ResNet-44, ResNet-56, and ResNet-110 on CIFAR-10.

Kernel	**Details of kernel function**
Gaussian	$K(x,y) = exp\left(-\frac{\|\|x-y\|\|^2}{2\sigma^2}\right)$
Laplacian	$K(x,y) = exp\left(-\frac{\|\|x-y\|\|}{\sigma}\right)$
Linear	$K(x,y) = x^T y + c$
Sigmoid	$K(x,y) = \tanh\left(ax^T + c\right)$

Table 3. Kernel functions of HSIC LASSO.

Model	**Kernel**	**Accuracy**	**FLOPs ↓**
VGG-16	Gaussian	93.83%	66%
	Laplacian	93.93%	66%
	Linear	92.39%	66%
	Sigmoid	86.13%	66%
ResNet-56	Gaussian	93.92%	54%
	Laplacian	93.99%	54%
	Linear	93.39%	54%
	Sigmoid	92.54%	54%

Table 4. Ablation study on kernel function.

ResNet-110 [17] on CIFAR-10 [26], with results presented in Fig. 3. We vary the sample sizes during the pruning stage to obtain the different compressed models, and then train these networks under the same settings. We can observe that as the sample size increases, the accuracy of pruned models becomes more stable. When the sample size is larger than 256, the performance of the pruned model is almost unchanged, indicating the stability of our APIB.

Selection of kernel. The kernel function of HSIC LASSO plays an important role for the results of pruning. In this paper, We compare some kernels listed in Tab.3 by conducting ablation experiments on the CIFAR-10 dataset using VGG16 and ResNet-56 to explore the impact of different kernels on pruned results. As shown in Tab. 4, The Gaussian and Laplacian kernels exhibit very similar accuracy, outperforming both the linear and sigmoid kernels.

Hypeparameter Ω selection. Ω decides the minimal number of channels in each layer after pruning. A high Ω reduces pruning potential and may even fail to achieve the target sparsity when the sparsity ratio is high, while a low Ω can cause excessive pruning and decreased performance. E.g., with a sparsity ratio of 95%, the VGG-16 achieves a 0.39% higher accuracy with a Ω of 5 compared to a Ω of 10, and a 0.4% higher accuracy compared to a Ω of 1. Similarly, for ResNet-56 with a pruning ratio of 85%, a Ω of 3 yields a 0.3% performance improvement compared to a Ω of 1. And a Ω of 5 doesn't achieve the desired sparsity ratio. Setting the Ω to 0 may result in complete pruning of a layer, causing layer collapse and rendering the model non-functional. For GoogleNet, the pruned model with a Ω of 9 has a better performance than those with higher or lower Ω at a sparsity of 77%. In conclusion, selecting the appropriate Ω actually depends on model size and sparsity ratio.

APIB vs. Other Criteria. In Sec. 3.4, we analyze Norm-based pruning from the standpoint of the IB theory and reveal that Norm-based criteria only focus on optimizing $HSIC(\tilde{X};Y)$, but ignore optimizing $HSIC(\tilde{X};X)$. Many results in Sec. 4 show that Norm-based pruning has a poor performance. We conduct an experiment to compare the channel selection results of different pruning criteria and calculate their similarity ratio in the same layer. In the i-th layer, the similarity ratio is defined as: $similarity\_ratio_i\left(A,B\right) = \frac{|A \cap B|}{|A|}$, where A is the pre-

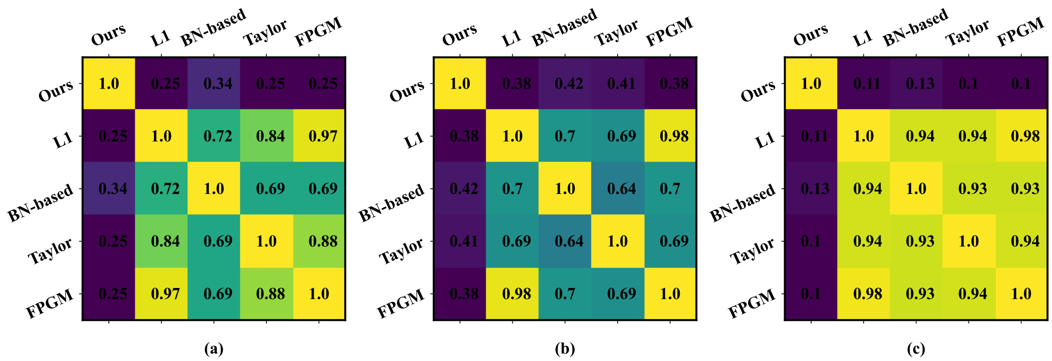

Figure 4. We compare different criteria in (a) 1-th convolution layer, (b) 8-th convolution layer and (c) 13-th convolution layer of VGG-16 on CIFAR-10. Deeper color represents a lower similarity ratio.

served filter set of method A; and B denotes the preserved filter set of method B, in the i-th layer. Fig. 4 shows the pruning results of the Bn-based and Taylor[41] are highly similar to the ones of Norm-based methods. However, there is a significant difference in chosen channels for APIB, which provides experimenta support for our analysis. However, the reason why the discarded filters of Taylor and BN-based methods are similar to the ones of Norm-based pruning, especially in shallower and deeper layers, is still unclear, which needs further studies in the future.

4.5. Extension to post-training pruning

We extend APIB to post-training pruning without fine-tuning. Compared to other baselines, APIB shows an obvious superiority. In Tab.5, at a sparsity level of 10%, pruned VGG-16 with APIB achieves a accuracy of 92.44%, which is significantly higher than L1, FPGM, and Hrank. At a sparsity level of 30%, pruned VGG16 with APIB achieves a higher accuracy of 91.84% compared to Hrank, while L1 and FPGM have already fallen below 30% accuracy.

Method	sparsity	accuracy
APIB	**10%**	**92.44**
Hrank	10%	86.98
L1	10%	83.18
FPGM	10%	84.80
APIB	**30%**	**91.84**
Hrank	30%	40.24

Table 5. The results of post-training pruning.

4.6. Extension to data-free pruning

we conducted experiments and found APIB can be extended as a data-free method by generating images randomly as inputs. For example, the pruned VGG16 achieved 94.04% accuracy on CIFAR-10 at a pruning rate of 60%, which is almost identical to the original result (94.08%). Similarly, the pruned ResNet50 achieved 70.76% accuracy on ImageNet at a pruning ratio of 76%, which is very close to the non-data-free result (70.67%).

4.7. Time cost comparison

Pruning with APIB takes several tens of seconds to a few minutes. APIB significantly reduces pruning time compared to Hrank and CHIP[50], which calculate rank or channel independence based on feature maps. The experimental results can be found in supplementary.

5. Conclusion

We propose a novel automatic pruning method called APIB that applies the Information Bottleneck (IB) principle to network pruning, achieving excellent performance on various benchmarks. Unlike previous heuristics-based pruning methods, APIB provides theoretical guidance for network pruning. Furthermore, we prove the equivalent relationship between HSIC Bottleneck and HSIC Lasso for the first time, which lays the foundation for utilizing HSIC Lasso to prune networks based on the IB principle. We also interpret Norm-based pruning from the perspective of Information Bottleneck, revealing its limitation. In the future, we plan to further investigate the relationship between other pruning criteria and APIB from the standpoint of information theory.

6. Acknowledgement

This work was supported by the National Key R&D Program of China (No.2022ZD0118202), the National Science Fund for Distinguished Young Scholars (No.62025603), the National Natural Science Foundation of China (No.U21B2037, No.U22B2051, No.62176222, No.62176223, No.62176226, No. 62072386, No.62072387, No.62072389, No.62002305 and No. 62272401), and the Natural Science Foundation of Fujian Province of China (No.2021J01002, No.2022J06001).

References

[1] Sungsoo Ahn, Shell Xu Hu, Andreas Damianou, Neil D Lawrence, and Zhenwen Dai. Variational information distillation for knowledge transfer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9163–9171, 2019.

[2] Manoj Alwani, Yang Wang, and Vashisht Madhavan. Decore: Deep compression with reinforcement learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12349–12359, 2022.

[3] Linhang Cai, Zhulin An, Chuanguang Yang, Yangchun Yan, and Yongjun Xu. Prior gradient mask guided pruning-aware fine-tuning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 1, 2022.

[4] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence*, 40(4):834–848, 2017.

[5] Wenlin Chen, James Wilson, Stephen Tyree, Kilian Weinberger, and Yixin Chen. Compressing neural networks with the hashing trick. In *International conference on machine learning*, pages 2285–2294. PMLR, 2015.

[6] Bin Dai, Chen Zhu, Baining Guo, and David Wipf. Compressing neural networks using the variational information bottleneck. In *International Conference on Machine Learning*, pages 1135–1144. PMLR, 2018.

[7] Xiaohan Ding, Xiangxin Zhou, Yuchen Guo, Jungong Han, Ji Liu, et al. Global sparse momentum sgd for pruning very deep neural networks. *Advances in Neural Information Processing Systems*, 32, 2019.

[8] Yuanzhi Duan, Yue Zhou, Peng He, Qiang Liu, Shukai Duan, and Xiaofang Hu. Network pruning via feature shift minimization. *arXiv preprint arXiv:2207.02632*, 2022.

[9] Sara Elkerdawy, Mostafa Elhoushi, Hong Zhang, and Nilanjan Ray. Fire together wire together: A dynamic pruning approach with self-supervised mask prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12454–12463, 2022.

[10] Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv preprint arXiv:1803.03635*, 2018.

[11] Shangqian Gao, Feihu Huang, Weidong Cai, and Heng Huang. Network pruning via performance maximization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9270–9280, 2021.

[12] Shangqian Gao, Feihu Huang, Jian Pei, and Heng Huang. Discrete model compression with resource constraint for deep neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1899–1908, 2020.

[13] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014.

[14] Arthur Gretton, Olivier Bousquet, Alex Smola, and Bernhard Schölkopf. Measuring statistical dependence with hilbert-schmidt norms. In *Algorithmic Learning Theory: 16th International Conference, ALT 2005, Singapore, October 8-11, 2005. Proceedings 16*, pages 63–77. Springer, 2005.

[15] Yi Guo, Huan Yuan, Jianchao Tan, Zhangyang Wang, Sen Yang, and Ji Liu. Gdp: Stabilized neural network pruning via gates with differentiable polarization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5239–5250, 2021.

[16] Song Han, Jeff Pool, John Tran, and William Dally. Learning both weights and connections for efficient neural network. *Advances in neural information processing systems*, 28, 2015.

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[18] Yang He, Ping Liu, Ziwei Wang, Zhilan Hu, and Yi Yang. Filter pruning via geometric median for deep convolutional neural networks acceleration. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4340–4349, 2019.

[19] Yang He, Ping Liu, Linchao Zhu, and Yi Yang. Filter pruning by switching to neighboring cnns with good attributes. *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

[20] Yihui He, Xiangyu Zhang, and Jian Sun. Channel pruning for accelerating very deep neural networks. In *Proceedings of the IEEE international conference on computer vision*, pages 1389–1397, 2017.

[21] Zhiqiang He, Yaguan Qian, Yuqi Wang, Bin Wang, Xiaohui Guan, Zhaoquan Gu, Xiang Ling, Shaoning Zeng, Haijiang Wang, and Wujie Zhou. Filter pruning via feature discrimination in deep neural networks. In *European Conference on Computer Vision*, pages 245–261. Springer, 2022.

[22] Geoffrey Hinton, Oriol Vinyals, Jeff Dean, et al. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2(7), 2015.

[23] Hengyuan Hu, Rui Peng, Yu-Wing Tai, and Chi-Keung Tang. Network trimming: A data-driven neuron pruning approach towards efficient deep architectures. *arXiv preprint arXiv:1607.03250*, 2016.

[24] Zhongzhan Huang, Wenqi Shao, Xinjiang Wang, Liang Lin, and Ping Luo. Rethinking the pruning criteria for convolutional neural network. *Advances in Neural Information Processing Systems*, 34:16305–16318, 2021.

[25] Donggyu Joo, Eojindl Yi, Sunghyun Baek, and Junmo Kim. Linearly replaceable filters for deep network channel pruning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 8021–8029, 2021.

[26] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[27] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 2015.

[28] Yann LeCun, John Denker, and Sara Solla. Optimal brain damage. *Advances in neural information processing systems*, 2, 1989.

[29] Namhoon Lee, Thalaiyasingam Ajanthan, and Philip HS Torr. Snip: Single-shot network pruning based on connection sensitivity. *arXiv preprint arXiv:1810.02340*, 2018.

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

[31] Yawei Li, Kamil Adamczewski, Wen Li, Shuhang Gu, Radu Timofte, and Luc Van Gool. Revisiting random channel pruning for neural network compression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 191–201, 2022.

[32] Yuchao Li, Shaohui Lin, Jianzhuang Liu, Qixiang Ye, Mengdi Wang, Fei Chao, Fan Yang, Jincheng Ma, Qi Tian, and Rongrong Ji. Towards compact cnns via collaborative compression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6438–6447, 2021.

[33] Mingbao Lin, Rongrong Ji, Shaojie Li, Yan Wang, Yongjian Wu, Feiyue Huang, and Qixiang Ye. Network pruning using adaptive exemplar filters. *IEEE Transactions on Neural Networks and Learning Systems*, 2021.

[34] Mingbao Lin, Rongrong Ji, Yan Wang, Yichen Zhang, Baochang Zhang, Yonghong Tian, and Ling Shao. Hrank: Filter pruning using high-rank feature map. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1529–1538, 2020.

[35] Congcong Liu and Huaming Wu. Channel pruning based on mean gradient for accelerating convolutional neural networks. *Signal Processing*, 156:84–91, 2019.

[36] 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.

[37] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015.

[38] Jian-Hao Luo and Jianxin Wu. An entropy-based pruning method for cnn compression. *arXiv preprint arXiv:1706.05791*, 2017.

[39] Jian-Hao Luo, Jianxin Wu, and Weiyao Lin. Thinet: A filter level pruning method for deep neural network compression. In *Proceedings of the IEEE international conference on computer vision*, pages 5058–5066, 2017.

[40] Wan-Duo Kurt Ma, JP Lewis, and W Bastiaan Kleijn. The hsic bottleneck: Deep learning without back-propagation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 5085–5092, 2020.

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

[42] Morten Østergaard Nielsen, Jan Østergaard, Jesper Jensen, and Zheng-Hua Tan. Compression of dnns using magnitude pruning and nonlinear information bottleneck training. In *2021 IEEE 31st International Workshop on Machine Learning for Signal Processing (MLSP)*, pages 1–6. IEEE, 2021.

[43] Roman Pogodin and Peter Latham. Kernelized information bottleneck leads to biologically plausible 3-factor hebbian learning in deep networks. *Advances in Neural Information Processing Systems*, 33:7296–7307, 2020.

[44] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28, 2015.

[45] Xiaofeng Ruan, Yufan Liu, Bing Li, Chunfeng Yuan, and Weiming Hu. Dpfps: dynamic and progressive filter pruning for compressing convolutional neural networks from scratch. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 2495–2503, 2021.

[46] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.

[47] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018.

[48] CH Sarvani, Mrinmoy Ghorai, Shiv Ram Dubey, and SH Shabbeer Basha. Hrel: Filter pruning based on high relevance between activation maps and class labels. *Neural Networks*, 147:186–197, 2022.

[49] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[50] Yang Sui, Miao Yin, Yi Xie, Huy Phan, Saman Aliari Zonouz, and Bo Yuan. Chip: Channel independence-based pruning for compact neural networks. *Advances in Neural Information Processing Systems*, 34:24604–24616, 2021.

[51] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[52] Yehui Tang, Yunhe Wang, Yixing Xu, Dacheng Tao, Chunjing Xu, Chao Xu, and Chang Xu. Scop: Scientific control for reliable neural network pruning. *Advances in Neural Information Processing Systems*, 33:10936–10947, 2020.

[53] Naftali Tishby, Fernando C Pereira, and William Bialek. The information bottleneck method. *arXiv preprint physics/0004057*, 2000.

[54] Naftali Tishby and Noga Zaslavsky. Deep learning and the information bottleneck principle. In *2015 ieee information theory workshop (itw)*, pages 1–5. IEEE, 2015.

[55] Joost van Amersfoort, Milad Alizadeh, Sebastian Farquhar, Nicholas Lane, and Yarin Gal. Single shot structured pruning before training. *arXiv preprint arXiv:2007.00389*, 2020.

[56] Chaoqi Wang, Guodong Zhang, and Roger Grosse. Picking winning tickets before training by preserving gradient flow. *arXiv preprint arXiv:2002.07376*, 2020.

[57] Jiaxing Wang, Haoli Bai, Jiaxiang Wu, Xupeng Shi, Junzhou Huang, Irwin King, Michael Lyu, and Jian Cheng. Revisiting parameter sharing for automatic neural channel number search. *Advances in Neural Information Processing Systems*, 33:5991–6002, 2020.

[58] Ying Wang, Yadong Lu, and Tijmen Blankevoort. Differentiable joint pruning and quantization for hardware efficiency. In *European Conference on Computer Vision*, pages 259–277. Springer, 2020.

[59] Zifeng Wang, Tong Jian, Aria Masoomi, Stratis Ioannidis, and Jennifer Dy. Revisiting hilbert-schmidt information bottleneck for adversarial robustness. *Advances in Neural Information Processing Systems*, 34:586–597, 2021.

[60] Zi Wang, Chengcheng Li, and Xiangyang Wang. Convolutional neural network pruning with structural redundancy reduction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14913–14922, 2021.

[61] Makoto Yamada, Wittawat Jitkrittum, Leonid Sigal, Eric P Xing, and Masashi Sugiyama. High-dimensional feature selection by feature-wise kernelized lasso. *Neural computation*, 26(1):185–207, 2014.

[62] Yangchun Yan, Rongzuo Guo, Chao Li, Kang Yang, and Yongjun Xu. Channel pruning via multi-criteria based on weight dependency. In *2021 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2021.

[63] Ruichi Yu, Ang Li, Chun-Fu Chen, Jui-Hsin Lai, Vlad I Morariu, Xintong Han, Mingfei Gao, Ching-Yung Lin, and Larry S Davis. Nisp: Pruning networks using neuron importance score propagation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 9194–9203, 2018.

[64] Yanfu Zhang, Shangqian Gao, and Heng Huang. Exploration and estimation for model compression. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 487–496, 2021.

[65] Yuxin Zhang, Mingbao Lin, Zhihang Lin, Yiting Luo, Ke Li, Fei Chao, Yongjian Wu, and Rongrong Ji. Learning best combination for efficient n: M sparsity. *arXiv preprint arXiv:2206.06662*, 2022.

[66] Xiawu Zheng, Yuexiao Ma, Teng Xi, Gang Zhang, Errui Ding, Yuchao Li, Jie Chen, Yonghong Tian, and Rongrong Ji. An information theory-inspired strategy for automatic network pruning. *arXiv preprint arXiv:2108.08532*, 2021.

[67] Aojun Zhou, Yukun Ma, Junnan Zhu, Jianbo Liu, Zhijie Zhang, Kun Yuan, Wenxiu Sun, and Hongsheng Li. Learning n: m fine-grained structured sparse neural networks from scratch. *arXiv preprint arXiv:2102.04010*, 2021.