SCIENCE CHINA
Information Sciences

• RESEARCH PAPER •

November 2026, Vol. 69, Iss. 11, 212104:1–212104:18
https://doi.org/10.1007/s11432-025-5096-8

Exploring dispersion of feature distribution from parameters for network pruning

Dongyue WU, Zilin GUO, Nong SANG & Changxin GAO*

National Key Laboratory of Multispectral Information Intelligent Processing Technology, School of Artifcial Intelligence and Automation, Huazhong University of Science and Technology, Wuhan 430074, China

Received 2 April 2025/Revised 13 August 2025/Accepted 18 December 2025/Published online 26 October 2026

Abstract Network pruning achieves significant advancements in deploying models on devices with constrained computational resources. Existing filter pruning methods adopt the criterion that features contributing less to the output are less important. However, they overlook the capability of features to capture crucial distinctions between various samples faithfully. In this paper, we employ the dispersion of feature distribution as an indicator of this capability and propose to preserve channels with large dispersion. The proposed distribution-based dispersion filter pruning method (DDisp) estimates the variance and coefficient of variation (CV) of each channel's feature distribution. To measure the dispersion, we exploit the approximated probability density of the distribution and establish mathematically grounded estimations towards the real variance and CV. Moreover, these estimations are built upon the stored parameters of normalization layers. This enables DDisp to circumvent the additional overhead inherent to data-driven methods, namely the costly network inference phase and the aggregation of actual statistics from a multitude of features. Extensive experiments demonstrate that DDisp consistently achieves excellent performance on various architectures, including ResNet, MobileNet, DenseNet, VGG, and ViT.

Keywords deep learning, model compression, filter pruning, feature distribution, dispersion

Citation Wu D Y, Guo Z L, Sang N, et al. Exploring dispersion of feature distribution from parameters for network pruning. Sci China Inf Sci, 2026, 69(11): 212104, https://doi.org/10.1007/s11432-025-5096-8

1 Introduction

Deep neural networks have shown superior performance in various computer vision applications, such as classification [1, 2], detection [3, 4], and segmentation [5]. However, their substantial overheads pose challenges to the deployment of high-performance models on edge devices with constrained resources. Thus, parameter quantization [6–11], low-rank approximation [12, 13], and network pruning [14–18] have been proposed to compress and accelerate models with high computational costs.

Network pruning can be categorized into two groups, namely weight pruning [19,20] and filter pruning [15,17,21]. Weight pruning eliminates connections to enlarge unstructured sparsity, which requires specialized libraries to facilitate accelerated sparse matrix computation. In contrast, filter pruning aims to remove the entire redundant filters to create a more compact model. This structured removal leads to the fact that the pruned network differs only in the channel numbers of each layer when compared to its pre-pruning state. Hence, filter pruning methods can achieve acceleration and parameter compression using off-the-shelf hardware without the need for specialized libraries.

Many existing filter pruning studies adhere to a contribution perspective, positing that filters associated with feature maps exerting a greater impact on the output should be retained, whereas those with lesser contributions ought to be pruned. Following this principle, numerous studies have proposed diverse metrics to measure the importance of filters. Some pioneering studies [21, 22] assessed the importance of individual filters based on their norms, wherein a larger norm signifies greater importance. Ref. [23] assumed that the contribution can be reflected by the back-propagated gradients of each filter. Recently, FPGM [24] and DepGraph [25] take advantage of the relationship between parallel filters and the dependencies in inter-layer computations to assess the contribution, respectively. Moreover, some data-driven methods quantify the contribution based on the feature distributions. For example, HRank [26] retains filters that contain more important information, as characterized by high rank. Similarly, based on the feature map matrices, CHIP [27] and FPC [28] use the sum of singular values to measure the

* Corresponding author (email: cgao@hust.edu.cn)

© Science China Press 2026

importance of the features extracted by each channel. APoZ [29] directly resorts to the distribution of the output feature values, pruning channels with a higher average percentage of zeros. SIRFP [18] exploits the contribution of the set of retained filters according to the accumulated spacial information of feature maps. Due to the reduced redundancy and improved generalization capacity, these contribution-based methods achieve significant success in network compression.

Nevertheless, we contend that channels with larger contributions in the well-trained network may not inherently provide abundant information for accurate prediction. On the one hand, certain common features shared across all samples may make substantial contributions to the output but offer scant information to distinguish different samples. On the other hand, certain features, despite not being particularly significant in terms of the magnitude of the output values or gradients, can more distinctly or comprehensively highlight subtle distinctions between different samples.

Building upon these considerations, we contend that filter pruning should originate from the essence of the classification task, preserving features that clearly exhibit inter-sample diversity and are more adept at distinctly differentiating between various samples. In this paper, we propose to employ high dispersion as an indicator of discriminative features and compress networks from a dispersion perspective. The proposed distribution-based dispersion filter pruning method (DDisp) measures the dispersion of feature distribution as the importance score. The variance and coefficient of variation (CV) are introduced to measure the absolute and relative variation of the feature distributions, which are then combined as our dispersion-based importance score. Furthermore, unlike those data-driven pruning methods [26, 27, 29] requiring an extra inference phase to obtain real statistics, we establish mathematically grounded estimations for the expectations and variance of the feature distribution. Beyond the commonly utilized scale factors and bias, DDisp also leverages the scarcely exploited running mean and running variance of batch normalization (BN) layers for the estimation of real statistics. Therefore, DDisp avoids the costly computation of inference on numerous real samples and collects those statistical values from the generated features. The proposed method can be easily deployed on various network architectures using BN and layer normalization (LN), including VGG [30], ResNet [1], MobileNetV2 [31], GoogleNet [32], DenseNet [33], and ViT [34]. Extensive experiments indicate that this mathematically grounded filter pruning method can achieve impressive pruning performance, especially at very high FLOPs reduction ratios. To summarize, our main contributions are as follows.

• From the dispersion perspective, we introduce a practical metric for filter importance based on the dispersion of feature distribution. Experimental findings illustrate that dispersion serves as an effective criterion for assessing filter importance, particularly under conditions of high acceleration ratio.

• To efficiently quantify the dispersion, we introduce a mathematical approach to estimate the dispersion statistics of feature distributions with the parameters of normalization layers.

• Extensive experimental results demonstrate that our method achieves a superior trade-off between accuracy and efficiency on various tasks across different popular network architectures.

2 Related work

The success of deep learning is inextricably linked to its substantial computational costs, which have spurred a spectrum of research into enhancing efficiency. These efforts span data efficiency [35–37], training efficiency [38, 39], and inference efficiency [6, 10]. Our work contributes to the last category, focusing specifically on structured network pruning, which will be elaborated in this section.

2.1 Structured pruning

Structured pruning aims to eliminate the entire filters (or channels, neurons, even layers, etc.), thereby rebuilding a lightweight model that retains the original architecture. Compared to unstructured pruning methods [19, 20], structured pruning methods [15, 17, 21] reduce memory consumption and speed up inference based on structural network sparsity, without requiring specialized hardware and libraries.

In the field of structured pruning, some approaches focus on removing channels according to the importance scores, while others attempt to train a sparse network with the help of regularization terms that penalize the weights of certain channels. In this study, we mainly focus on the importance criteria for ranking and pruning the channels. Existing pruning criteria can be categorized into two groups: weight-dependent and data-driven criteria. Although they differ from each other in the reliance on input data for importance calculation, most studies of both types follow the same idea: channels (or filters) that contribute significantly to the output features should

be retained, while those with smaller contributions should be removed. As far as we know, existing studies have overlooked whether different samples can be clearly distinguished in the feature space after pruning.

2.2 Weight-dependent pruning criteria

Weight-dependent importance metrics only utilize the learned parameters of models to quantify the channel-wise importance. Compared with data-driven methods, weight-dependent methods do not involve an inference process on input data, thus usually requiring lower computational costs.

Pioneering structural pruning methods often employ the norm of filters as the importance metric. For example, Li et al. [21] assumed that filters with larger ℓ_1-norms can generate stronger activation maps, consequently making larger contributions to the final predictions. Building on this idea, SFP [22] argues that ℓ_2-norm is more effective than ℓ_1-norm in practice. Similarly, early methods such as Network Sliming [40] believe larger scaling factors in the BN layers suggest a greater influence on the subsequent features. Following Network Sliming, some studies also leverage BN parameters to evaluate channel-wise contribution. BNFI [41] adopts the "larger-absolute-value-greater-impact" principle and estimates the expectation of the activation as the importance score based on the scale factors and bias of BN layers. Variational CNN pruning [42] introduces the channel saliency for BN layers to indicate importance. Channels with lower saliency values are prioritized for pruning.

However, according to FPGM [24], this "smaller-norm-less-important" assumption is not always effective. Ref. [43] pointed out that kernel-norm-based channel importance lacks spatial saliency; thus the proposed SPSRC reformulates convolutions as matrix multiplications and uses the singular values of the transformed matrices as scores. FPGM instead proposes removing filters with high redundancy, which is quantified by the distance between each filter and the geometric median. The closer a filter is to the geometric median, the less distinctive and discriminative features it can capture, thus contributing less to the final prediction. Inspired by FPGM, RED [44] and RED++ [45] explore to merge similar filters, using the uneven depth-wise separation and input-wise splitting techniques to prune filters, respectively. C-SGD [46] organizes redundancy during training by making some filters identical, enabling harmless filter removal without fine-tuning. Moreover, COP [47] extends the similarity-based importance score within layers to a cross-layer criterion.

2.3 Data-driven pruning criteria

The importance scores of data-driven methods heavily rely on input data. Usually, the calculation of data-driven criteria necessitates that models perform training or inference on batches of training data.

Some data-driven studies introduce new contribution indicators that are dependent on feature maps. NISP [48] employs Inf-FS [49] to rank the final features, and then the importance scores are back-propagated to previous layers. APOZ [29] prunes channels with a large average percentage of zeros (APoZ), indicating that only a small fraction of elements in the activation maps are activated, and thus minimally contribute to the succeeding feature maps. HRank [26] resorts to removing channels with lower ranks of feature maps. CHIP [27] employs the nuclear norm as a soft metric to prune channels, compared to the hard rank of features. Similarly, FPC [28] also performs SVD on the feature maps to obtain their singular values, and proposes using the sum of the singular values of the feature matrix to represent the importance of the corresponding filter. The rationale behind them lies in that higher rank and nuclear norms signify richer information. Other studies manage to control the reconstruction error (RE), which actually roughly equates to pruning features with smaller absolute values. LASSO [50] seeks to minimize the RE between the original and pruned features by solving the LASSO regression problem. Similarly, AOFP [51] proposes to minimize the RE of the $(\ell + 1)$-th layer's output for pruning channels within the ℓ-th layer during training.

Despite their performance advantages, these methods still assess the importance of channels based on their contribution to the network's final output. Moreover, the costs associated with inference on real data, additional memory for feature storage, and the time required for processing and metric calculation remain barriers to the broader implementation of these data-driven methods.

2.4 Correlation with other norm-based methods

Our proposed method shares similarities with other approaches also based on Norm layers, including Network Sliming [40] and BNFI [41]. Like these methods, DDisp also utilizes the stored parameters in Norm layers to construct the importance function. However, there are two key differences between our DDisp and these approaches. Firstly, both Network Sliming and BNFI construct their importance functions based on the contribution to the output, directly using scale factors or estimating the mean values of output features. In contrast, our method posits

that features that exhibit greater variations among different samples are those containing more discriminative characteristics. Consequently, we base our importance function on the divergence of feature map distributions across samples. Furthermore, while Network Sliming and BNFI solely employ the scale factor and bias to assess the importance of BN output features, our approach further incorporates running mean and running variance. This integration enables the direct measurement of the importance of features at the BN input side, namely the output features of the Conv layer. In this sense, our method offers a holistic and comprehensive importance assessment of the Conv-BN module as a unified entity.

3 Methodology

In this section, we elaborate on the proposed DDisp. Before we go to our method, we firstly introduce the problem definition and the background knowledge. After that, we present the ideal importance function from the proposed dispersion perspective. Then, we show how to estimate the statistics of feature distribution based on the parameters of normalization layers. The real statistics are replaced by the corresponding approximations as a feasible solution for distribution-based filter pruning without the extra inference computational costs.

3.1 Problem definition

Given a trained neural network and all its layers $\mathcal{L} = \{l^1, l^2, l^3, \ldots, l^K\}$, filter pruning aims to identify and remove filters with minimal impact on accuracy loss. To achieve better accuracy under a certain given prune ratio, we assess the importance of each filter and selectively eliminate the least crucial ones. This optimization problem can be formulated as follows:

$$\min_{\delta_{ij}} \sum_{i=1}^{K} \sum_{j=1}^{N_i} \delta_{ij} \Phi(l^i, j) \quad \text{s.t.} \sum_{j=1}^{N_i} \delta_{ij} = n_i, \tag{1}$$

where $\Phi(\cdot,\cdot)$ is the importance function, $\Phi(l^i, j)$ denotes the importance score of the j-th filter in the i-th layer l^i, $\delta_{ij} \in 0, 1$ signifies the pruning decision for a filter, where $\delta_{ij} = 1$ indicates the filter is selected for pruning. N_i is the number of filters in l^i, and n_i is the number of filters to be pruned in l^i. Therefore, the core of filter pruning methodologies involves developing systematic and effective importance functions to guide pruning decisions, preferably those underpinned by transparent mathematical foundations.

Most contribution-based filter pruning methods employ a data-independent approach to estimate the importance scores only based on the parameters stored in the trained parameters $\mathcal{W} = \{W^1, W^2, \ldots, W^K\}$. Due to the unavailability of real distribution information for each feature map, these methods may result in suboptimal accuracy and compression performance. On the contrary, some data-driven methods [26, 27, 29] compute the statistics for importance score calculation based on the actual features $\mathcal{F} = \{F^1, F^2, F^3, \ldots, F^K\}$. Although effective, they necessitate the initial inference on real input data to obtain $\mathcal{F}$ and subsequently collect statistics from it, resulting in a cumbersome pruning process with additional computational labor and time.

In this paper, we propose a feasible filter pruning approach that evaluates importance based on the statistics of feature distribution using solely stored parameters. Our method is based on mathematical derivations to approximate the statistics of feature distribution. Using these approximate statistics, we compute the dispersion of feature maps as importance scores. It is worth noting that, in addition to scale factors and bias, we leverage the running mean and running variance, which have been scarcely explored in the field of filter pruning, to obtain accurate statistics.

3.2 Background knowledge

Before introducing our approach, let us first review the widely used BN layer and the law of the unconscious statistician (LOTUS).

Batch normalization layer. BN layer adopts the moving average manner to estimate the mean and variance of the real feature distribution. These estimates, known as running mean and running variance, are stored as parameters to normalize the feature during inference. The output feature generation of the j-th channel of a BN layer during inference can be formulated as

$$y_j^k = \mathrm{BN}_{\gamma_j, \beta_j}(x_j^k) = \gamma_j \frac{x_j^k - \hat{\mu}_j}{\sqrt{(\hat{\sigma}_j)^2 + \epsilon}} + \beta_j, \tag{2}$$

where $y_j^k \in \mathbb{R}$ is the k-th generated output element of the j-th channel output feature map $\mathbf{Y}_j \in \mathbb{R}^{b\times h\times w}$, $x_j^k \in \mathbb{R}$ denotes the k-th element in the j-th channel input feature map $\mathbf{X}_j \in \mathbb{R}^{b\times h\times w}$. γ_j and β_j are the learned scale factor and bias, respectively. $\hat{\mu}_j$ and $(\hat{\sigma}_j)^2$ are the running mean and running variance, respectively. We can posit that the input features approximately conform to a normal distribution $N(\mu, \sigma^2)$. Similar assumptions or approximations are widely adopted by deep learning studies, like BatchNorm [52], VAE [53], SNN [54], and StyleGAN [55]. In such a scenario, the feed-forward process of BN can be interpreted as: (1) standardizing the input distribution to the standard normal distribution $N(0, 1)$ and subsequently (2) transforming the standard normal distribution to the output distribution $N(\beta_j, \gamma_i^2)$.

Law of the unconscious statistician. In statistics, the law of the unconscious statistician describes the expected value of a function $g(X)$, where X is a random variable [56]. For a continuous distribution of X with its probability density function $p_X(x)$, the expected value $\mathbb{E}[g(X)]$ of $g(X)$ is

$$\mathbb{E}[g(X)] = \int_{-\infty}^{+\infty} p_X(x) \cdot g(x)\ \mathrm{d}x. \tag{3}$$

3.3 Dispersion importance function

Within the framework of (1), we propose to measure the dispersion of the feature distribution as the importance score for each filter. Our rationale is that if the projections in the feature space of different samples distinctly represent their original differences, then the corresponding filters play a crucial role in the final prediction. Conversely, if various samples with substantial differences are closely situated in a feature space, the corresponding filter fails to capture discriminative information, making it challenging to distinguish between them.

The standard deviation is commonly employed for quantifying the dispersion. It reveals the scale of absolute variation, which is associated with the capability to clearly distinguish between different samples. However, given the substantial disparity in mean values and ranges across feature maps, relying solely on the absolute metric lacks rationality. It is imperative to consider the relative variability of the data, facilitating a more intuitive and meaningful comparison among distributions with differing mean values. Thus, we introduce the CV as a relative metric

$$\mathrm{CV}(X) = \frac{\sqrt{\mathbb{V}(X)}}{\mathbb{E}(X)}, \tag{4}$$

where $X \sim P(X)$ signifies a random variable which conforms to the distribution $P(X)$, $\mathbb{V}(X)$ denotes the variance, and $\mathbb{E}(X)$ denotes the expectation (mean) of $P(X)$. Since the CV generally applies when the mean is greater than zero, in the implementation, if the expectation is less than zero, we take the absolute value of a to calculate the coefficient of variation.

To comprehensively measure the importance of each filter in the j-th convolution layer, we define our dispersion score of the feature distribution $\phi(P(O_j^i))$ using both variance and CV

$$\phi(P(O_j^i)) = \sqrt{\mathbb{V}(O_j^i)} \cdot \mathrm{CV}(O_j^i), \tag{5}$$

where $O_j^i \sim P(O_j^i)$ represents the random variable that conforms to the output feature distribution of the j-th channel in the i-th convolution (or linear) layer. Aside from this intuitively explanation, we also prove that CV is related to the upper bound of information entropy of channels in Subsection 3.4, which could provide a theoretical explanation for introducing CV. Preserving channels with large CV will lead to a small upper bound of information entropy. The combination of standard deviation and CV can provide rich discriminative information while avoiding an undesired extremely high degree of uncertainty.

Please note that the dispersion of the convolution output feature distribution can merely measure the importance of intermediate features (the output of convolution layers) for a commonly used Conv-Norm-AF (activation function) module. In fact, the activation maps of the module directly influence the subsequent modules. However, Eq. (5) only measures the quality of intermediate features generated by convolution layers, ignoring the quality of activation maps, which have direct influences on the subsequent layers. Inspired by this thought, we further incorporate the dispersion score of the output activation maps into the importance score. Combining the dispersion of intermediate features and the final activation maps together, we finally propose the importance function for the j-th filter in the i-th Conv-Norm-AF module

$$\Phi(l^i, j) = \phi(P(O_j^i))\phi(P(A_j^i)), \tag{6}$$

where $A_j^i \sim P(A_j^i)$ denotes the random variable conforming to the distribution of the output activation maps $P(A_j^i)$. We adopt a multiplicative form to combine $\phi(P(O_j^i))$ and $\phi(P(A_j^i))$. This design emphasizes filters that exhibit

high dispersion in both stages, ensuring that the retained channels are consistently discriminative throughout the Conv-Norm-Activation forward pass.

3.4 Theoretical explanation to minimizing dispersion

In the above section, we intuitively explain that CV is a relative dispersion indicator that helps to form a comprehensive comparison of the dispersion of different distributions. Here, we also provide the theoretical elaboration that CV is related to the upper bound of the information entropy of features.

Let us rethink the optimization goal of network pruning based on the information entropy of features. For an image classification task, the optimization goal is to reduce the cross entropy between the ground truth and the prediction generated by the deep neural network. Hence, it is natural to have the thought that the forward process of a neural network can be regarded as an entropy-reducing procedure the classification network filters out the noisy information to reduce the final entropy of prediction. For the feature maps of numerous hidden layers in deep learning networks, high information entropy of a feature map also indicates the underside high degree of uncertainty. Thus, in each layer, if we aim at preserving features (and its corresponding filters) with less information entropy, the optimization issue can be formulated as

$$\min_{\delta_{ij}} \sum_{i=1}^{K} \sum_{j=1}^{N_i} \delta_{ij} H(l^i, j) \quad \text{s.t.} \sum_{j=1}^{N_i} \delta_{ij} = n_i, \tag{7}$$

where $H(l^i, j)$ is the information entropy of the feature distribution of the j-th channel in i-th layer l^i

$$H(l^i, j) = \int_{-\infty}^{+\infty} -p_X(x) \log[p_X(x)] \, \mathrm{d}x. \tag{8}$$

Taking the activation maps of a Conv-BN-ReLU module as an example, we try to calculate the information entropy. Following the assumption that each input feature map of ReLU roughly conforms to a normal distribution $N(\beta_j, \gamma_j^2)$, the information entropy of the output activation $Y = \mathrm{ReLU}(X)$ can be formulated as follows:

$$\begin{aligned} H(l^i, j) &= \int_{-\infty}^{+\infty} -p_Y(y) \log[p_Y(y)] \, \mathrm{d}y = \int_{0}^{+\infty} -p_X(x) \log[p_X(x)] \mathrm{d}x \ - P(X \leqslant 0) \log[P(X \leqslant 0)] \\ &= \frac{1}{4}[\log(2\pi\gamma_j^2) + 1] \left[1 + \mathrm{erf}\left(\frac{\beta_j}{\sqrt{2}\gamma_j}\right)\right] - \frac{\beta_j \mathrm{e}^{-\frac{\beta_j^2}{2\gamma_j^2}}}{2\sqrt{2\pi}\gamma_j} - \frac{1}{2}\mathrm{erfc}\left(\frac{\beta_j}{\sqrt{2}\gamma_j}\right) \log\left[\frac{1}{2}\mathrm{erfc}\left(\frac{\beta_j}{\sqrt{2}\gamma_j}\right)\right] \\ &\leqslant \frac{1}{4}[\log(2\pi\gamma_j^2) + 1] \left[1 + \mathrm{erf}\left(\frac{\beta_j}{\sqrt{2}\gamma_j}\right)\right] - \frac{\beta_j \mathrm{e}^{-\frac{\beta_j^2}{2\gamma_j^2}}}{2\sqrt{2\pi}\gamma_j} + \frac{1}{\mathrm{e}}, \end{aligned} \tag{9}$$

where the inequality is based on the fact that $x\log(x) \leqslant \mathrm{e}^{-1}$ for $\forall x > 0$. Let $s_j^i = \frac{\beta_j}{\sqrt{2}\gamma_j}$, then $H_{\mathrm{up}}(s_j^i)$ is an upper bound of $H(l^i, j)$

$$H(l^i, j) \leqslant H_{\mathrm{up}}(s_j^i) = \frac{1 + C}{4} \left[1 + \mathrm{erf}\left(s_j^i\right)\right] - \frac{s_j^i \mathrm{e}^{-(s_j^i)^2}}{2\sqrt{\pi}} + \frac{1}{\mathrm{e}}, \tag{10}$$

where $C = \log(2\pi \cdot \max\{\frac{1}{2\pi}, \gamma_1^2, \gamma_2^2, \ldots, \gamma_n^2\}) \geqslant 0$ is a constant for each $H_{\mathrm{up}}(s_j^i)$. The derivative $\frac{\partial H_{\mathrm{up}}}{\partial s_j^i}$ is formulated as

$$\frac{\partial H_{\mathrm{up}}}{\partial s_j^i} = \frac{\mathrm{e}^{-(s_j^i)^2} \left[2(s_j^i)^2 + C\right]}{2\sqrt{\pi}}. \tag{11}$$

Thus, $\frac{\partial H_{\mathrm{up}}}{\partial s_j^i} \geqslant 0$ is true for $s \in \mathbb{R}$, which means $H_{\mathrm{up}}(s_j^i)$ is monotonically increasing. Consequently, smaller s_j^i leads to smaller $H_{\mathrm{up}}(s_j^i)$, indicating lower value of information entropy. Therefore, we can keep those channels with smaller s_j^i to retain smaller upper bound of the information entropy. Moreover, since the activation features after ReLU function have a bigger expectation and less data variation (all values less than zero are projected to zero by ReLU), $H_{\mathrm{up}}(\frac{\mathbb{E}(A_j^i)}{\sqrt{2\mathbb{V}(A_j^i)}})$ is also a looser upper bound of $H(l^i, j)$. Therefore, we can optimize the following problem

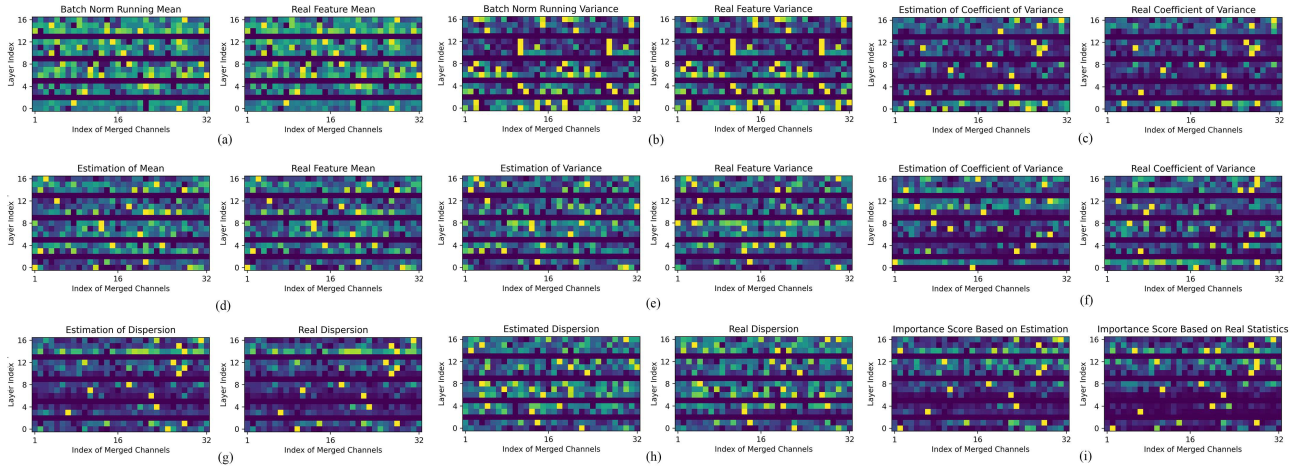


Figure 1 Visualization of our derived estimations and the real statistics of feature distributions on VGG-16. DS denotes the dispersion score. (a) Mean of intermediate features; (b) variance of intermediate features; (c) CV of intermediate features; (d) mean of activation; (e) variance of Activation; (f) CV of activation; (g) DS of intermediate features; (h) DS of activation; (i) importance score. In each subplot, our estimations and the real statistics are shown on the left and right, respectively. Intermediate features denote the output of convolution layers, and Activation denotes the output of activation functions. We first normalize the values of all channels within each layer, then use various colors to denote different values. For clarity, all layers are averaged to 32 channels. The results demonstrate that our estimations are consistent with the real collected statistics.

to maintain the low information entropy upper bound of both the intermediate features and activation maps in the pruned network

$$\min_{\delta_{ij}} \sum_{i=1}^{K} \sum_{j=1}^{N_i} \delta_{ij} H_{\text{up}} \left(\frac{\mathbb{E}(A_j^i)}{\sqrt{2\mathbb{V}(A_j^i)}} \right) \quad \text{s.t.} \sum_{j=1}^{N_i} \delta_{ij} = n_i, \tag{12}$$

thus, the optimization problem is equivalent to finding the filters with top-n_j smallest $\frac{\mathbb{E}(A_j^i)}{\sqrt{\mathbb{V}(A_j^i)}}$, namely the filters with top-n_j biggest CV of activation maps. This reflects why we introduce the CV into the proposed dispersion-based importance score. However, merely minimizing the information entropy could lead to a bad case where all retained channels have few pixels bigger than zero in the activation maps after the ReLU function. In this case, the information entropy is minimized, but there is no discrimination between all pixels in the activation map. Thus, we combine the standard deviation with CV for a trade-off between better discrimination of features and less uncertainty by retaining channels with large dispersion and maintaining low information entropy at the same time.

3.5 Distribution estimation of the intermediate features

Collecting real statistics can be done by running the trained model over all the training samples. Nevertheless, this procedure requires massive computational costs when compared with data-independent filter pruning methods. Even though existing data-driven methods manage to reduce the amount of required input data (over ten thousand in [29], hundreds in HRank [26] and CHIP [27]), substantial amounts of time are still needed for the importance computation. Hence, the natural inclination is to resort to approximations in place of real statistics.

Fortunately, we have observed that the readily available running mean and running variance stored in the BN layers serve as accurate approximations of the actual mean and variance, respectively. Figures 1(a) and (b) illustrate that the running mean and running variance can accurately estimate the mean value (expectation) and variance of the true intermediate feature distribution in Conv-BN-AF modules. Furthermore, our estimations for the activation distribution (refer to Subsection 3.6) can also faithfully reflect the relative magnitudes of statistics across different channels. This discovery is sufficient to indicate that expending only a minimal cost on a mathematically grounded derivation can facilitate the accurate estimation of the statistics.

Based on these observations, we can conclude that the running mean and running variance are quite precise estimations of the expectation and variance. Hence, we can substitute $\phi(P(O_j^i))$ with an estimation $\hat{\phi}(P(O_j^i))$ without the real features

$$\phi(P(O_j^i)) \approx \hat{\phi}(P(O_j^i)) = \frac{(\hat{\sigma}_j^i)^2}{\hat{\mu}_j^i}, \tag{13}$$

where $\hat{\mu}_j^i$ and $(\hat{\sigma}_j^i)^2$ denote the j-th channel's running mean and running variance stored in the subsequent BN layer of the i-th convolution layer.

3.6 Distribution estimation of the activation

Considering the input variable X of the activation function $f(\cdot)$, namely the variable sampled from the j-th channel output distribution of the BN layer, it also roughly exhibits a normal distribution $N(\beta_j, \gamma_j^2)$ (refer to Subsection 3.2). Thus, the probability density function of variable X can be formulated as

$$p_X(x) = \frac{1}{\gamma_j\sqrt{2\pi}} \mathrm{e}^{-\frac{(x-\beta_j)^2}{2\gamma_j^2}}. \tag{14}$$

Based on the LOTUS expressed in (3), we derive an estimated mean value of the output activation distribution as

$$\hat{\mathbb{E}}(A_j^i) = \int_{-\infty}^{+\infty} p_Y(y) \cdot y \ \mathrm{d}y = \int_{-\infty}^{+\infty} p_X(x) \cdot f(x) \ \mathrm{d}x, \tag{15}$$

where $Y = f(X)$ is the random variable that conforms to the output activation distribution of j-th channel $P(A_j^i)$, and $p_Y(\cdot)$ denotes the probability density function of Y. Similarly, we can also estimate the variance of the output activation distribution

$$\hat{\mathbb{V}}(A_j^i) = \int_{-\infty}^{+\infty} p_X(x)[f(x) - \hat{\mathbb{E}}(A_j^i)]^2 \ \mathrm{d}x. \tag{16}$$

To avoid the large computational cost of inference to obtain the real statistics, we can employ the approximation $\hat{\phi}(P(A_j^i))$ as a substitution for $\phi(P(A_j^i))$

$$\phi(P(A_j^i)) \approx \hat{\phi}(P(A_j^i)) = \frac{\hat{\mathbb{V}}(A_j^i)}{\hat{\mathbb{E}}(A_j^i)}. \tag{17}$$

For simplicity, we take the ReLU function as an example of the activation function. Thus, the estimation $\hat{\mathbb{E}}(A_j^i)_{\mathrm{relu}}$ of the output activation map of a commonly used Conv-BN-ReLU module is formulated as

$$\hat{\mathbb{E}}(A_j^i)_{\mathrm{relu}} = \int_{-\infty}^{+\infty} \frac{\mathrm{ReLU}(x)}{\gamma_j\sqrt{2\pi}} \ \mathrm{e}^{-\frac{(x-\beta_j)^2}{2\gamma_j^2}} \mathrm{d}x = \int_{0}^{+\infty} \frac{x}{\gamma_j\sqrt{2\pi}} \ \mathrm{e}^{-\frac{(x-\beta_j)^2}{2\gamma_j^2}} \ \mathrm{d}x. \tag{18}$$

Based on (18), let $t = \frac{x-\beta_j}{\sqrt{2}\gamma_j}$, we have the following transformation:

$$\begin{aligned} \hat{\mathbb{E}}(A_j^i)_{\mathrm{relu}} =& \int_{\frac{-\beta_j}{\sqrt{2}\gamma_j}}^{+\infty} \frac{\beta_j}{\sqrt{\pi}} \ \mathrm{e}^{-t^2} \ \mathrm{d}t + \int_{\frac{-\beta_j}{\sqrt{2}\gamma_j}}^{+\infty} \frac{t}{\sqrt{\pi}} \ \mathrm{e}^{-t^2} \sqrt{2}\gamma_j \ \mathrm{d}t \\ =& \frac{\beta_j}{2}\left[1 + \mathrm{erf}\left(\frac{\beta_j}{\sqrt{2}\gamma_j}\right)\right] + \frac{\gamma_j}{\sqrt{2\pi}} \mathrm{e}^{\frac{-\beta_j^2}{2\gamma_j^2}} = \frac{\beta_j}{2}(1+A) + B, \end{aligned} \tag{19}$$

where $A = \mathrm{erf}(\frac{\beta_j}{\sqrt{2}\gamma_j})$, $B = \frac{\gamma_j}{\sqrt{2\pi}} \mathrm{e}^{\frac{-\beta_j^2}{2\gamma_j^2}}$, and $\mathrm{erf}(\cdot)$ is the Gauss error function

$$\mathrm{erf}(z) = \frac{2}{\sqrt{\pi}} \int_0^z \mathrm{e}^{-t^2} \ \mathrm{d}t. \tag{20}$$

The fourth equation in (19) is based on the fact that $\mathrm{erf}(-x) = -\mathrm{erf}(x)$, Since many libraries [57] support the efficient calculation of $\mathrm{erf}(\cdot)$, we can estimate the $\hat{\mathbb{E}}(A_j^i)$ at low computational cost and fast speed. In contrast, previous methods based on real statistics have to bear the extra memory and computation cost to forward on a large number of samples, store the features, and calculate the real statistics.

Using the same substitution method as (19), $\hat{\mathbb{V}}(A_j^i)_{\mathrm{relu}}$ can be formulated as

$$\hat{\mathbb{V}}(A_j^i)_{\mathrm{relu}} = \frac{\hat{\mathbb{E}}(A_j^i)^2}{2}(1-A) + \left[\beta_j - 2\hat{\mathbb{E}}(A_j^i)\right] B + \frac{1}{2}\left([\hat{\mathbb{E}}(A_j^i) - \beta_j]^2 + \gamma_j^2\right)(1+A). \tag{21}$$

The detailed proof of (21) is shown in the Supporting information.

3.7 Feasible distribution-based pruning solution

For each Conv-BN-ReLU module, we can calculate the importance score based on the estimations

$$\Phi(l^i, j) \approx \hat{\phi}(P(O_j^i))\hat{\phi}(P(A_j^i)). \tag{22}$$

Combining this derived approximation with (1), our final optimization objective is as follows:

$$\min_{\delta_{ij}} \sum_{i=1}^{K} \sum_{j=1}^{N_i} \delta_{ij} \frac{(\hat{\sigma}_j^i)^2}{\hat{\mu}_j^i} \frac{\hat{\mathbb{V}}(A_j^i)}{\hat{\mathbb{E}}(A_j^i)} \qquad \text{s.t.} \sum_{j=1}^{N_i} \delta_{ij} = n_i. \tag{23}$$

This final objective of filter pruning can be easily applied to most CNNs consisting of Conv-BN-ReLU modules. Addressing this optimization problem does not necessitate gathering all features for statistics; solely the stored parameters of BN are sufficient. For the importance functions of modules with different activation functions and architectures, such as the Tanh activation function, ReLU6 function in MobileNetV2 [31], modules using LN in the ViT [34], and modules using group normalization (GN) and instance normalization (IN), please refer to the Supporting information.

4 Experiments

4.1 Datasets and implementation details

Datasets and metrics. This paper mainly focuses on image classification tasks. We conduct extensive experiments on CIFAR-10, CIFAR-100 [58] and ImageNet [59]. CIFAR-10 has 10 classes, containing 50k images for training and 10k images for performance evaluation with 32×32 resolution. CIFAR-100 is similar to CIFAR-10 but categorized into 100 classes. ImageNet is a large-scale and challenging dataset containing 1.28 million training images and 50k validation images of 1000 classes. We evaluate the proposed method on various widely-used neural networks. We provide the experimental results of ResNet [1], VGG [30], DenseNet [33], MobileNet [31], and ViT [34]. Moreover, we also evaluate the effectiveness of DDisp on semantic segmentation and object detection tasks. We prune the classic and popular segmentation network DeeplabV3 [60] on the Cityscapes [61] and ADE20k [62] datasets. Experiments of pruning object detection networks SSD and Mask R-CNN on COCO2017 dataset [63] are also provided.

To evaluate the pruning performance for classification, we adopt the top-1 accuracy (Acc) and the accuracy drop, which is defined as $\Delta\text{Acc} = \text{Acc(pruned)} - \text{Acc(base)}$. The floating point operations (FLOPs) and FLOPs speed-up ratio are also adopted to measure the acceleration performance of DDisp, namely FLOPs(base)/FLOPs(pruned). We also use the number of parameters (Params) to show the model size of the pruned models. The mean intersection over union (mIoU) is adopted for semantic segmentation evaluation on the Cityscapes and ADE20K. On COCO2017, we employ the average precision (AP) for performance of different object detection models.

Pruning details. We perform filter pruning in a post-training manner. The trained base models are first pruned and then fine-tuned to recover the accuracy. Following the settings of ResRep [64], Greg [65], and DepGraph [25], we also deploy a sparse training stage before pruning on CIFAR-10 and CIFAR-100 to promote sparsity during the training. This could help to retain the accuracy performance since the base model already learned how to use fewer channels to understand images and generate accurate predictions. However, we do not adopt sparse training for ImageNet considering the large machine-time consumption. The Torchvision models are directly pruned and fine-tuned in our ImageNet experiments. In addition, we also adopt the dynamic pruning setting (denoted as DDisp-dyn) on ImageNet to further facilitate the performance of our method. Following previous studies [66], we train and prune the model from scratch in a single phase. We prune the model during training every 2 epochs until the 180th epoch. The pruned model continues training until the 205th epoch.

4.2 Visualization of the estimated statistics

Since our method depends on the derived estimations of the feature distribution, we first check the accuracy of those estimations. We compare the estimated values with the real statistics collected from the real feature distributions of both the intermediate feature maps and activation maps, namely the input and output of BN, respectively. For VGG-16, since there are too many channels (up to 512) in each layer, we reduce the channel numbers for visualization to 32 for easier and clearer comparison. We compute the statistics for each channel, then average the values of adjacent channels to form a total of 32 such averaged values, which are subsequently normalized across channels and visualized in Figure 1. The lighter the color, the larger the magnitude of values.

The comparison between Figures 1(a) and (b) demonstrates that the stored running mean and running variance can provide quite accurate approximations of the real average and variance of the intermediate feature distributions. Figures 1(c) and (g) show that the estimated CV and dispersion score exhibit high consistency with the real values as well. Figures 1(d)–(f) and (h) demonstrate that there is only negligible disparity between the estimations and real statistics for activation maps. Consequently, the calculated importance scores using our approximations are also generally consistent with those based on the real statistics, as shown in Figure 1(i).

Despite being implemented in a "data-free" manner, the proposed method for estimating distribution information is surprisingly effective. These high-quality estimations also corroborate the rationality of our method and its assumption that features roughly conform to normal distributions. We believe that this method may contribute to the study of feature distribution in deep neural networks and elucidate the process of feature learning as well.

4.3 Results of image classification

Results on CIFAR. We evaluate DDisp against other state-of-the-art methods in Table 1. When pruning the ResNet-56 model trained on the CIFAR-10 dataset, the proposed DDisp reduces 52.96% FLOPs but still improves the top-1 accuracy by 0.83. In addition, DDisp achieves a higher ratio of pruned FLOPs with far less accuracy drop. The pruning performance of our proposed DDisp is competitive and comparable to the performance of other methods adopting fine-grained layer-wise pruning ratios and powerful sparsity regularization terms.

More importantly, the comparison between DDisp and other data-driven methods that also exploit distribution information, like HRank [26] and CHIP [27], demonstrates that even without the actual statistics collected from the real feature maps, pruning filters with our mathematically founded estimations can still generate a slimmer and even more accurate pruned model. This phenomenon indicates the effectiveness of both the proposed method to estimate statistics and the specially designed importance function that reflects the dispersion of feature distributions.

Results on ImageNet. We also report the pruning results on ImageNet, including ResNet, MobileNet, and ViT in Table 2. We use "dyn" to denote the dynamic pruning setting. Dynamic pruning is a popular technique that prunes the network every few epochs during training, which is also called adaptive pruning. Similar settings are commonly adopted by various structural pruning methods [24,50,67,68]. The results demonstrate that our proposed method achieves satisfactory results on various popular network architectures. The success of DDisp on ImageNet further substantiates the correlation between the large dispersion of feature distributions and the commendable performance.

4.4 Results of downstream tasks

Results on semantic segmentation. We conduct pruning experiments on the popular and challenging Cityscapes and ADE20K datasets. Notably, all reported results in Table 3 of other methods are based on reproduced implementations since most methods except DCFP [69] primarily focus on the image classification task. On both datasets, DDisp demonstrates superior performance by producing pruned models with fewer FLOPs and higher mIoU scores. Despite the complexity and substantial sample volume of these datasets, DDisp still attains remarkable results. Additionally, given that each pixel in the images is a sample for segmentation, datasets with higher image resolution contain significantly more samples than those used for classification (like ImageNet-1K). Meanwhile, the original colors and textures of various objects and their corresponding pixels also vary considerably, even within the same class, resulting in increased variation and dispersion. DDisp achieves a notable performance advantage over the contribution-based methods, such as FPGM and DepGraph. These comparisons underscore the superiority of DDisp in contexts where samples exhibit large mutual differences in the feature space. In addition, although DCFP is specially designed to address the long-tail distribution issue in segmentation, our DDisp, without such task-specific optimization, still outperforms it, demonstrating its effectiveness in semantic segmentation.

Results on object detection. The results of object detection are presented in Table 4. To ensure a fair comparison, we adopt the same training and pruning settings as those of CHEX [66]. The baseline results are the reported results from the original papers. We also reproduce the FPGM [24] results for comparative analysis. The proposed DDisp achieves a great improvement in terms of the AP_L and AP_{75} performance over FPGM, demonstrating enhanced capability in locating large objects. The pruned models may ignore the noisy features and focus on those large objects with richer contextual information. Additionally, the performance gain over the baseline models illustrates that DDisp effectively reduces the redundancy of neural networks while maintaining robust detection performance in various scenarios.

Table 1 Pruning results on CIFAR-10 and CIFAR-100. The proposed DDisp achieves state-of-the-art results compared with other pruning methods. * denotes our reproduced results. Bold values indicate the best results.

	Method	Base	Pruned	ΔAcc.	FLOPs	Params
CIFAR-10/ ResNet-56	Base	93.53	–	–	127.12M	0.85M
	L1 [21]	93.04	93.06	+0.02	90.90M	0.73M
	CFDP [70]	93.26	93.97	+0.71	90.35M	0.66M
	HRank [26]	93.26	93.52	+0.26	88.72M	0.71M
	NISP [48]	93.04	93.01	−0.03	81.00M	0.49M
	SR-GKP [71]	–	94.08	–	71.86M	0.48M
	CHIP [27]	93.26	94.16	**+0.90**	65.94M	0.48M
	LASSO [50]	93.26	91.87	−1.00	65.94M	–
	WL50 [72]	93.26	92.68	−0.58	64.83M	–
	JSATF [73]	94.17	93.71	−0.46	61.10M	–
	DepGraph [25]	93.53	93.77	+0.24	60.13M	–
	DPFPS [74]	93.81	93.20	−0.61	59.92M	0.45M
	DSP [75]	93.26	94.08	+0.82	59.89M	0.36M
	MFP [76]	93.59	93.56	−0.03	59.40M	–
	LRFP [77]	93.59	93.57	−0.02	59.40M	–
	BN-SFP [22]	93.59	93.29	−0.30	59.40M	–
	FPC [28]	93.78	94.01	+0.23	81.53M	0.29M
	ARPruning [78]	93.26	93.67	+0.41	66.70M	0.49M
	UDSP [79]	93.12	93.78	+0.66	63.56M	0.43M
	IFSO [80]	93.68	93.65	−0.03	63.56M	–
	SPM [81]	–	93.50	–	63.20M	–
	GlobalPru [82]	93.50	93.49	−0.01	60.21M	0.17M
	CSD* [83]	93.53	93.86	+0.33	59.69M	0.45M
	DDisp (ours)	93.53	**94.35**	+0.82	**59.38M**	0.29M
	AFP [84]	93.93	92.94	−0.99	55.50M	–
	GAL [85]	93.26	91.58	−1.68	49.99M	0.29M
	DepGraph [25]	93.53	93.64	+0.11	49.85M	–
	REPrune [86]	–	93.39	–	50.48M	–
	SPSRC* [43]	93.59	92.77	−0.82	50.23M	0.36M
	C-SGD [87]	93.39	93.62	+0.23	49.77M	0.33M
	DDisp (ours)	93.53	**93.88**	**+0.35**	**49.73M**	0.23M
	DTP [88]	93.36	92.46	−0.90	35.41M	–
	HRank [26]	93.26	90.72	−2.54	32.52M	0.27M
	CHIP [27]	93.26	92.05	−1.21	34.79M	0.24M
	ASTER [89]	93.80	92.79	−1.01	33.00M	–
	DDisp (ours)	93.53	**92.95**	**−0.58**	**31.29M**	0.16M
CIFAR-100/ VGG-19	Base	73.50	–	–	512.73M	20.1M
	Kron-OBD [90]	73.34	60.70	−12.64	89.47M	–
	EigenD [90]	73.34	65.18	−8.16	58.28M	–
	GReg-1 [65]	74.02	67.55	−6.47	58.26M	6.03M
	GReg-2 [65]	74.02	67.75	−6.27	58.00M	6.03M
	DTP [88]	74.02	67.74	−6.28	57.94M	–
	DepGraph [25]	73.50	70.39	−3.11	56.72M	–
	DDisp (ours)	73.50	**71.11**	**−2.39**	**56.64M**	5.94M

4.5 Ablation study

Contribution-based vs. dispersion-based importance functions. To demonstrate the efficacy of the dispersion-based importance function, we compare its pruning performance against contribution-based importance functions. We employ the L1-Norm and introduce a D-Mean, which is the product of the estimated mean value (expectation) of both intermediate and activation feature distributions. These two importance criteria serve as representatives of contribution-based methods. The results on VGG-19 and ResNet-56 presented in Figure 2 clearly indicate that DDisp outperforms both L1-Norm and D-Mean, which supports our assumption that channels with

Table 2 Pruning results on ImageNet. Under different FLOPs settings, DDisp can achieve competitive pruning performance. * denotes our reproduced results.

	Method	Base	Pruned	ΔAcc.	FLOPs
ResNet-50	Base	76.15	–	–	4.12G
	SSS [91]	76.12	74.18	−1.94	2.82G
	GReg-2 [65]	76.13	75.36	−0.77	2.77G
	TPP [92]	76.13	75.60	−0.50	2.77G
	AOFP-C1 [51]	75.34	75.63	+0.29	2.58G
	ABCPruner	76.15	74.84	−0.31	2.45G
	ThiNet [14]	72.88	72.04	−0.84	2.44G
	SFP [22]	76.15	74.61	−1.54	2.40G
	FPGM [24]	76.15	75.50	−0.65	2.38G
	BNFI [41]	76.33	75.47	−0.86	2.34G
	HRank [26]	76.15	74.98	−1.17	2.30G
	Meta [93]	76.60	75.20	−1.40	2.30G
	Taylor [17]	76.18	74.50	−1.68	2.25G
	IFSO [80]	76.55	75.70	−0.85	2.47G
	FPC* [28]	76.15	75.25	−0.90	2.29G
	SPSRC* [43]	76.15	75.40	−0.75	2.28G
	DDisp (ours)	76.15	**76.06**	−0.09	**2.24G**
	CCP [94]	76.15	75.50	−0.65	2.11G
	GFP* [95]	76.15	75.44	−0.71	2.04G
	DepGraph* [25]	76.15	75.28	−0.87	2.04G
	IFSO [80]	76.55	75.08	−1.43	2.06G
	DDisp (ours)	76.15	75.89	−0.36	**1.97G**
	DDisp-dyn (ours)	–	**77.04**	–	1.98G
	BNFI [41]	76.33	75.02	−1.31	1.94G
	FPGM [24]	76.15	74.83	−1.32	1.92G
	MFP [76]	76.15	74.86	−1.29	1.91G
	Hinge [96]	76.10	74.70	−1.40	1.88G
	DCP [15]	76.01	74.95	−1.06	1.83G
	DAIS [97]	76.15	74.45	−1.70	1.83G
	ARPruning [78]	76.15	72.31	−3.84	**1.80G**
	DDisp (ours)	76.15	**75.43**	−0.74	**1.80G**
	HRank [26]	76.15	71.98	−4.17	1.56G
	PFR [98]	–	74.40	–	1.40G
	CHIP [27]	76.15	73.30	−2.85	0.96G
	DDisp (ours)	76.15	73.90	−2.69	0.96G
	DDisp-dyn (ours)	–	**74.44**	–	**0.96G**
Mob-V2	Base	71.87	–	–	0.32G
	AutoSlim [99]	71.80	67.90	−3.90	0.15G
	DepGraph* [25]	71.87	68.07	−3.80	0.15G
	Meta [93]	74.70	68.20	−6.50	**0.14G**
	DDisp (ours)	71.87	**68.36**	−3.51	**0.14G**
ViT-B/16	Base	81.07	–	–	17.59G
	CP-ViT [100]	77.91	77.38	−0.53	14.70G
	DepGraph* [25]	81.07	77.72	−3.35	9.92G
	DDisp (ours)	81.07	**78.02**	−3.05	**9.90G**

large contributions may not inherently lead to accurate prediction.

Dispersion of intermediate and activation features. We investigate whether incorporating the dispersion of both intermediate and activation feature distributions is warranted for importance calculation. The DDisp w/o Int and DDisp w/o Act criteria are introduced, which denote ignoring the dispersion score of intermediate features and activation maps, respectively. The comparison in Figure 3 leads to the conclusion that incorporating both the dispersion of the intermediate features and activation maps instead of only one of them is more powerful. Moreover, although the activation maps directly affect the subsequent layers, ensuring that the intermediate features with

Table 3 Comparison of semantic segmentation performance on DeeplabV3-ResNet50 network.

Dataset	Method	mIoU	FLOPs
Cityscapes	Base	81.6	1418.3G
	Slimming [40]	79.9	575.2G
	FPGM [24]	79.1	553.8G
	Taylor [17]	80.3	564.9G
	DepGraph [25]	80.0	561.9G
	DCFP [69]	80.7	555.7G
	DDisp (ours)	81.0	554.1G
ADE20K	Base	45.3	177.4G
	Slimming [40]	40.6	72.3G
	FPGM [24]	39.8	72.4G
	Taylor [17]	40.8	73.8G
	DepGraph [25]	41.0	70.3G
	DCFP [69]	44.8	71.1G
	DDisp (ours)	44.9	68.6G

Table 4 Object detection pruning results on COCO2017 validation set. Our method achieves the best trade-off between FLOPs and AP. * denotes our reproduction.

Method	FLOPs speedup	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
SSD ResNet50							
Base	**1.0**×	25.2	42.7	25.8	7.3	27.1	40.8
DMCP [101]	**2.0**×	24.1	41.2	24.7	6.7	25.6	39.2
FPGM* [24]	**2.0**×	25.2	42.4	25.9	7.0	27.2	41.3
CHEX-1 [66]	**2.0**×	25.9	43.0	26.8	7.8	27.8	41.7
REPrune [86]	**2.0**×	25.0	42.3	25.9	7.4	26.8	40.3
DDisp (ours)	**2.0**×	**26.4**	**43.9**	**27.2**	**7.9**	**28.6**	**43.3**
Mask R-CNN ResNet50							
Base	**1.0**×	37.3	**59.0**	40.2	21.9	**40.9**	48.1
DMCP* [101]	**2.0**×	36.4	57.8	39.7	21.0	38.7	47.6
FPGM* [24]	**2.0**×	36.5	57.8	39.9	21.3	38.6	47.7
CHEX [66]	**2.0**×	37.3	58.5	40.4	21.7	39.0	49.5
DDisp (ours)	**2.0**×	**37.5**	58.8	**40.7**	**22.0**	39.8	**49.8**

Table 5 Ablation on the form of combination of the dispersion scores of intermediate features and activation maps.

Combination	$\Phi(l^i, j)$	FLOPs	Pruned	ΔAcc.
Baseline (Acc. 73.50 FLOPs 512.73M)				
Additive	$\hat{\phi}(P(O_j^i)) + \hat{\phi}(P(A_j^i))$	56.64M	70.74	−2.76
Weighted (5:1)	$\hat{\phi}(P(O_j^i)) + \frac{1}{5}\hat{\phi}(P(A_j^i))$	56.64M	70.69	−2.81
Weighted (1:5)	$\frac{1}{5}\hat{\phi}(P(O_j^i)) + \hat{\phi}(P(A_j^i))$	56.64M	70.72	−2.78
Multiplicative	$\hat{\phi}(P(O_j^i))\hat{\phi}(P(A_j^i))$	56.64M	**71.11**	**−2.39**
Additive	$\hat{\phi}(P(O_j^i)) + \hat{\phi}(P(A_j^i))$	32.05M	67.43	−6.07
Weighted (5:1)	$\hat{\phi}(P(O_j^i)) + \frac{1}{5}\hat{\phi}(P(A_j^i))$	32.05M	67.27	−6.23
Weighted (1:5)	$\frac{1}{5}\hat{\phi}(P(O_j^i)) + \hat{\phi}(P(A_j^i))$	32.05M	67.54	−5.96
Multiplicative	$\hat{\phi}(P(O_j^i))\hat{\phi}(P(A_j^i))$	32.05M	**68.01**	**−5.49**

higher dispersion can also offer an improvement. We believe this phenomenon can be attributed to the fact that the combination of both of them can ensure the consistency of features before and after the activation function, potentially leading to a more stable feature representation.

Analysis of the combination between the intermediate and activation dispersion scores. We conduct ablation studies in Table 5 to explore the better combination form of $\phi(P(O_j^i))$ and $\phi(P(A_j^i))$ in (6). The multiplicative form ensures that the retained channels can consistently highlight the distinction between samples in both intermediate (Int) and activation (Act) maps. However, additive or weighted-sum combinations may preserve filters with high dispersion in only one stage, which could propagate noise or redundant features. For example, when all

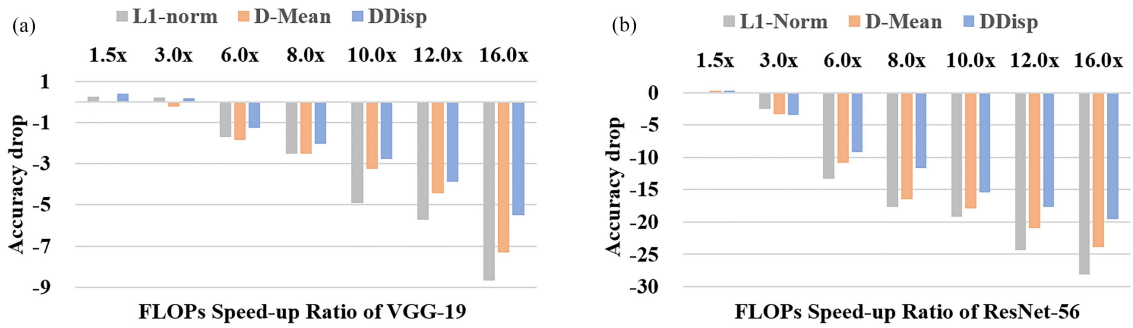


Figure 2 Ablation study on contribution (L1-Norm, D-Mean) and dispersion (DDisp) importance on CIFAR-100. D-Mean denotes the estimated mean value. (a) Results on VGG-19; (b) results on ResNet-56.

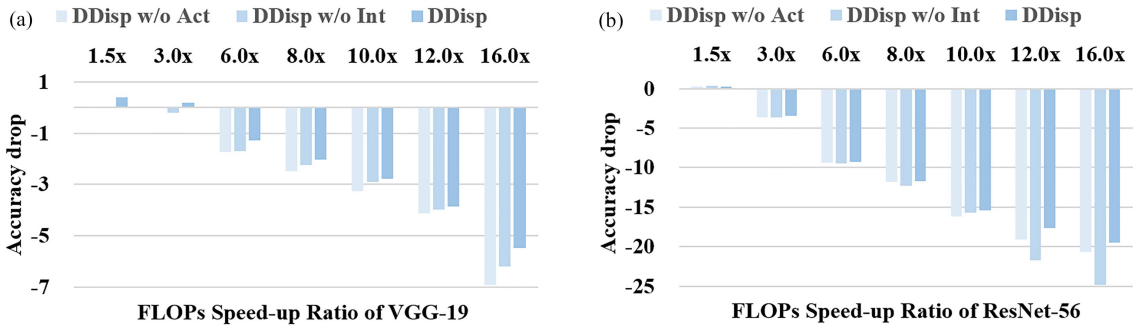


Figure 3 Ablation study on intermediate and activation features on CIFAR-100. w/o Act and w/o Int denotes that the $\phi(P(A_j^i))$ and $\phi(P(O_j^i))$ are ignored, respectively. (a) Results on VGG-19; (b) results on ResNet-56.

Int features are negative, the Act after ReLU may contain only zeros for all samples. Nevertheless, the additive form may still retain such channels due to their large dispersion of the Int, even though they cannot distinguish different samples.

Performance on different FLOPs speedup ratios. The results presented in Table 6 and Figure 2 reveal an intriguing observation: our method exhibits more substantial improvements over L1-Norm on large FLOPs speedup ratios (e.g., 10.0×–16.0×) compared to smaller ratios (1.5×–3.0×). A plausible explanation is that under configurations with low FLOPs reduction, the pruned networks can still generate informative representations through a significant number of remaining channels. However, in scenarios with large FLOPs speedup ratios, where only a limited number of channels survive, the richness of information for each channel becomes crucial. In such cases, our dispersion importance function excels at preserving channels with significant inter-sample diversity, enhancing the distinctiveness between samples and facilitating easier classification. This finding could serve as inspiration for future research on filter pruning, particularly under large pruning ratios. The DDisp-pruned networks maintain full-precision parameters; integrating quantization with DDisp could provide additional FLOPs reduction and efficiency improvements. In this case, DDisp serves as a structural compression step, while quantization methods [10, 11] further reduce arithmetic and memory cost, forming a comprehensive compression pipeline.

Performance on various networks. We additionally perform pruning experiments on various backbones including ResNet, VGGNet, GoogleNet, DenseNet, MobileNetV2, and ViT. DDisp shows compatibility on networks with either dense connections or parallel configurations, as seen in DenseNet. Results in Tables 3 and 4 demonstrate that DDisp also performs well on different downstream network architectures for segmentation and detection. The outcomes affirm the generalization capability of DDisp across diverse and intricate architectures.

Ablation on the batch size. Since our method is founded on the parameters of normalization layers, we conduct experiments to investigate the performance of DDisp with varying batch sizes during the training of unpruned models and pruned models. The results in Table 7 show that the batch size indeed affects the ΔAcc. Nevertheless, DDisp achieves consistent significant performance improvements over L1-Norm and DepGraph across different batch size configurations. This finding suggests that DDisp is more robust to variations in batch size compared to other pruning methods.

Table 6 Ablation study on CIFAR-100 for different network architectures and FLOPs configurations.

Model	Importance function	Pruned accuracy under different FLOPs speedup ratio						
		1.5×	3.0×	6.0×	8.0×	10×	12×	16×
ResNet-56 (72.53)	Random	70.85 (−1.68)	67.84 (−4.69)	60.35 (−12.18)	54.88 (−17.65)	51.59 (−20.94)	46.54 (−25.99)	39.29 (−33.24)
	L1-Norm	72.41 (−0.12)	**69.95 (−2.58)**	59.16 (−13.37)	54.90 (−17.63)	53.29 (−19.24)	48.20 (−24.33)	44.38 (−28.15)
	DDisp	**72.57 (+0.04)**	68.60 (−3.93)	**61.43 (−11.10)**	**59.88 (−12.65)**	**57.1 (−15.43)**	**54.92 (−17.61)**	**52.99 (−19.54)**
VGG-19 (73.50)	Random	71.49 (−2.01)	68.52 (−4.98)	62.49 (−11.01)	59.56 (−13.94)	60.12 (−13.38)	58.39 (−15.11)	54.86 (−18.64)
	L1-Norm	73.76 (+0.26)	**73.71 (+0.21)**	71.78 (−1.72)	71.00 (−2.50)	68.59 (−4.91)	67.76 (−5.74)	64.81 (−8.69)
	DDisp	**73.88 (+0.38)**	73.68 (+0.18)	**72.22 (−1.28)**	**71.48 (−2.02)**	**70.72 (−2.78)**	**69.63 (−3.87)**	**68.01 (−5.49)**
VGG-16 (73.65)	Random	73.35 (−0.30)	72.16 (−1.49)	67.89 (−5.76)	65.96 (−7.69)	64.73 (−8.92)	63.88 (−9.77)	61.03 (−12.62)
	L1-Norm	73.80 (+0.15)	72.75 (−0.90)	70.48 (−3.17)	68.55 (−5.10)	66.47 (−7.18)	65.78 (−7.87)	63.32 (−10.33)
	DDisp	**73.93 (+0.28)**	**73.01 (−0.64)**	**71.14 (−2.51)**	**69.38 (−4.27)**	**67.78 (−5.87)**	**66.23 (−7.42)**	**63.77 (−9.88)**
GoogleNet (77.71)	Random	76.34 (−1.37)	76.11 (−1.60)	72.45 (−5.26)	69.62 (−8.09)	69.18 (−8.53)	68.06 (−9.65)	67.42 (−10.29)
	L1-Norm	77.18 (−0.53)	76.98 (−0.73)	75.55 (−2.16)	73.55 (−4.16)	72.28 (−5.43)	69.01 (−8.70)	67.73 (−9.98)
	DDisp	**77.89 (+0.18)**	**77.49 (−0.22)**	**75.98 (−1.73)**	**74.50 (−3.21)**	**73.19 (−4.52)**	**73.03 (−4.68)**	**72.30 (−5.41)**
DenseNet-121 (78.50)	Random	78.62 (+0.12)	75.02 (−3.48)	70.42 (−8.08)	69.27 (−9.23)	66.54 (−11.96)	66.45 (−12.05)	64.33 (−14.17)
	L1-Norm	78.82 (+0.32)	78.92 (+0.42)	78.26 (−0.24)	77.37 (−1.13)	76.98 (−1.52)	75.26 (−3.24)	69.39 (−9.11)
	DDisp	**79.19 (+0.69)**	**79.54 (+1.04)**	**78.49 (−0.01)**	**77.77 (−0.73)**	**77.08 (−1.42)**	**75.80 (−2.70)**	**72.50 (−6.00)**
MobileNetV2 (70.55)	Random	70.61 (+0.06)	64.37 (−6.18)	59.26 (−11.29)	55.44 (−15.11)	53.13 (−17.42)	49.53 (−21.02)	44.86 (−25.69)
	L1-Norm	71.05 (+0.50)	71.01 (+0.46)	69.97 (−0.58)	68.40 (−2.15)	67.35 (−3.20)	65.78 (−4.77)	62.01 (−8.54)
	DDisp	**71.22 (+0.67)**	**71.54 (+0.99)**	**70.21 (−0.34)**	**68.80 (−1.75)**	**67.78 (−2.77)**	**66.32 (−4.23)**	**62.62 (−7.93)**

Table 7 Ablation on different batch size. The batch size is set to the same for both the baseline model and the pruned model.

Method	Batch size	Base	Pruned	ΔAcc.	FLOPs
L1-Norm	64	72.62	66.08	−6.54	57.24M
DepGraph			67.09	−5.53	57.99M
DDisp			**69.09**	**−3.53**	**56.54M**
L1-Norm	128	73.50	69.02	−4.48	56.84M
DepGraph			70.39	−3.11	56.72M
DDisp			**71.11**	**−2.39**	**56.64M**
L1-Norm	256	73.81	70.01	−3.80	57.09M
DepGraph			70.43	−3.38	57.86M
DDisp			**71.12**	**−2.69**	**56.75M**

5 Limitation

While our method can generalize to different activation functions, the estimation formulation for specific functions necessitates mathematical derivation. Although popular math tools, such as Mathematica, can assist in this derivation, a solid foundation in statistics and calculus remains essential. Furthermore, DDisp shows moderate effectiveness under small pruning ratios. This limitation may be attributed to the retention of a substantial number of filters, which enables the continued capture of rich discriminative information and maintains a significant discrepancy between samples in the feature space even after low pruning ratio compression.

6 Conclusion

In this study, we point out that existing structural pruning methods ignore the ability of remaining channels to faithfully reflect discriminative distinctions between different samples in the feature space. Therefore, we present DDisp, a method that quantifies the dispersion of the feature distribution as filter importance. Moreover, we also introduce a feasible approach to accurately estimate the statistics of real feature distributions. DDisp is compatible with various popular networks on classification, segmentation, and detection tasks, showing competitive pruning performance. Specifically, DDisp exhibits outstanding performance on large prune ratios.

Acknowledgements This work was supported by National Natural Science Foundation of China (Grant No. 62176097) and Hubei Provincial Natural Science Foundation of China (Grant No. 2022CFA055).

Supporting information Appendixes A–E. The supporting information is available online at info.scichina.com and link.springer.com. The supporting materials are published as submitted, without typesetting or editing. The responsibility for scientific accuracy and content remains entirely with the authors.

References

1 He K, Zhang X, Ren S, et al. Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016. 770–778
2 Tan M, Le Q. Efficientnet: Rethinking model scaling for convolutional neural networks. In: Proceedings of the 36th International Conference on Machine Learning, 2019. 6105–6114
3 Girshick R. Fast R-CNN. In: Proceedings of the IEEE International Conference on Computer Vision, 2015. 1440–1448
4 Ren S, He K, Girshick R, et al. Faster R-CNN: towards real-time object detection with region proposal networks. IEEE Trans Pattern Anal Mach Intell, 2017, 39: 1137–1149
5 Long J, Shelhamer E, Darrell T. Fully convolutional networks for semantic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015. 3431–3440
6 Han S, Mao H, Dally W J. Deep compression: compressing deep neural networks with pruning, trained quantization and Huffman coding. ArXiv: 1510.00149
7 Xu Y, Wang Y, Zhou A, et al. Deep neural network compression with single and multiple level quantization. In: Proceedings of the AAAI Conference on Artificial Intelligence, 2018
8 Esser S K, McKinstry J L, Bablani D, et al. Learned step size quantization. In: Proceedings of the International Conference on Learning Representations, 2020
9 Jacob B, Kligys S, Chen B, et al. Quantization and training of neural networks for efficient integer-arithmetic-only inference. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018. 2704–2713
10 Qin H, Liu X, Ma X, et al. BiVM: accurate binarized neural network for efficient video matting. IEEE Trans Pattern Anal Mach Intell, 2025, 47: 9250–9265
11 Qin H, Ding Y, Zhang X, et al. Diverse sample generation: pushing the limit of generative data-free quantization. IEEE Trans Pattern Anal Mach Intell, 2023, 45: 11689–11706
12 Yang Z, Wang Y, Liu C, et al. LegoNet: Efficient convolutional neural networks with lego filters. In: Proceedings of the International Conference on Machine Learning, 2019. 7005–7014
13 Yu X, Liu T, Wang X, et al. On compressing deep models by low rank and sparse decomposition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017. 7370–7379
14 Luo J H, Wu J, Lin W. ThiNet: A filter level pruning method for deep neural network compression. In: Proceedings of the IEEE International Conference on Computer Vision, 2017. 5058–5066
15 Zhuang Z, Tan M, Zhuang B, et al. Discrimination-aware channel pruning for deep neural networks. In: Proceedings of the 32nd Conference on Neural Information Processing Systems (NeurIPS 2018), 2018. 31
16 Lin T, Stich S, Barba L, et al. Dynamic model pruning with feedback. In: Proceedings of the International Conference on Learning Representations, 2020
17 Molchanov P, Mallya A, Tyree S, et al. Importance estimation for neural network pruning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019. 11264–11272
18 Wu D, Guo Z, Yu L, et al. Structural pruning via spatial-aware information redundancy for semantic segmentation. In: Proceedings of the AAAI Conference on Artificial Intelligence, 2025. 8368–8376
19 Han S, Pool J, Tran J, et al. Learning both weights and connections for efficient neural network. In: Proceedings of the 29th International Conference on Neural Information Processing Systems, 2015. 28
20 Carreira-Perpinán M A, Idelbayev Y. "Learning-Compression" algorithms for neural net pruning. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018. 8532–8541
21 Li H, Kadav A, Durdanovic I, et al. Pruning filters for efficient convnets. ArXiv: 1608.08710
22 He Y, Kang G, Dong X, et al. Soft filter pruning for accelerating deep convolutional neural networks. In: Proceedings of the International Joint Conference on Artificial Intelligence, 2018
23 Molchanov P, Tyree S, Karras T, et al. Pruning convolutional neural networks for resource efficient inference. ArXiv: 1611.06440
24 He Y, Liu P, Wang Z, et al. Filter pruning via geometric median for deep convolutional neural networks acceleration. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019. 4340–4349
25 Fang G, Ma X, Song M, et al. DepGraph: towards any structural pruning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2023. 16091–16101
26 Lin M, Ji R, Wang Y, et al. HRank: filter pruning using high-rank feature map. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020. 1529–1538
27 Sui Y, Yin M, Xie Y, et al. CHIP: Channel independence-based pruning for compact neural networks. In: Proceedings of the 35th Conference on Neural Information Processing Systems (NeurIPS 2021), 2021. 34: 24604–24616
28 Chen Y, Wen X, Zhang Y, et al. FPC: filter pruning via the contribution of output feature map for deep convolutional neural networks acceleration. Knowl-Based Syst, 2022, 238: 107876
29 Hu H, Peng R, Tai Y W, et al. Network trimming: a data-driven neuron pruning approach towards efficient deep architectures. ArXiv: 1607.03250
30 Simonyan K, Zisserman A. Very deep convolutional networks for large-scale image recognition. ArXiv: 1409.1556
31 Sandler M, Howard A, Zhu M, et al. MobileNetV2: inverted residuals and linear bottlenecks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018. 4510–4520
32 Szegedy C, Liu W, Jia Y, et al. Going deeper with convolutions. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015. 1–9
33 Huang G, Liu Z, van der Maaten L, et al. Densely connected convolutional networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017. 4700–4708
34 Dosovitskiy A, Beyer L, Kolesnikov A, et al. An image is worth 16×16 words: transformers for image recognition at scale. ArXiv: 2010.11929
35 Li Y, Han H, Shan S, et al. DISC: learning from noisy labels via dynamic instance-specific selection and correction. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2023. 24070–24079
36 Zheng R, Han C, Gao C, et al. Linear feature source prediction and recombination network for noisy label learning. IEEE Trans Circuits Syst Video Technol, 2025, 35: 1648–1659
37 Guo Z, Wu D, Gao C, et al. Selective multiple classifiers for weakly supervised semantic segmentation. CAAI Trans Intel Tech, 2025, 10: 1688–1702
38 Micikevicius P, Narang S, Alben J, et al. Mixed precision training. In: Proceedings of the International Conference on Learning Representations, 2018
39 Wu D, Guo Z, Zuo J, et al. Partial forward blocking: a novel data pruning paradigm for lossless training acceleration. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2025. 319–328
40 Liu Z, Li J, Shen Z, et al. Learning efficient convolutional networks through network slimming. In: Proceedings of the IEEE International Conference on Computer Vision, 2017. 2736–2744

41 Oh J, Kim H, Baik S, et al. Batch normalization tells you which filter is important. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, 2022. 2645–2654
42 Zhao C, Ni B, Zhang J, et al. Variational convolutional neural network pruning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019. 2780–2789
43 Sun X, Shi H. Towards better structured pruning saliency by reorganizing convolution. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, 2024. 2204–2214
44 YVINEC E, Dapogny A, Cord M, et al. RED: looking for redundancies for data-freestructured compression of deep neural networks. In: Proceedings of the 35th Conference on Neural Information Processing Systems (NeurIPS 2021), 2021. 20863–20873
45 Yvinec E, Dapogny A, Cord M, et al. RED++: data-free pruning of deep neural networks via input splitting and output merging. IEEE Trans Pattern Anal Mach Intell, 2022, 45: 3664–3676
46 Hao T, Ding X, Han J, et al. Manipulating identical filter redundancy for efficient pruning on deep and complicated CNN. IEEE Trans Neural Netw Learn Syst, 2024, 35: 16831–16844
47 Wang W, Fu C, Guo J, et al. COP: customized deep model compression via regularized correlation-based filter-level pruning. In: Proceedings of the International Joint Conference on Artificial Intelligence, 2019. 3785–3791
48 Yu R, Li A, Chen C F, et al. NISP: pruning networks using neuron importance score propagation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2018. 9194–9203
49 Roffo G, Melzi S, Cristani M. Infinite feature selection. In: Proceedings of the IEEE International Conference on Computer Vision, 2015. 4202–4210
50 He Y, Zhang X, Sun J. Channel pruning for accelerating very deep neural networks. In: Proceedings of the IEEE International Conference on Computer Vision, 2017. 1389–1397
51 Ding X, Ding G, Guo Y, et al. Approximated oracle filter pruning for destructive CNN width optimization. In: Proceedings of the International Conference on Machine Learning, 2019. 1607–1616
52 Ioffe S, Szegedy C. Batch normalization: accelerating deep network training by reducing internal covariate shift. In: Proceedings of the International Conference on Machine Learning, 2015. 448–456
53 Kingma D P, Welling M. Auto-encoding variational bayes. ArXiv: 1312.6114
54 Klambauer G, Unterthiner T, Mayr A, et al. Self-normalizing neural networks. In: Proceedings of the 31st International Conference on Neural Information Processing Systems, 2017. 30
55 Karras T, Laine S, Aila T. A style-based generator architecture for generative adversarial networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019. 4401–4410
56 Schervish M J, DeGroot M H. Probability and Statistics. Harlow: Pearson Education Limited. 2014
57 Paszke A, Gross S, Massa F, et al. PyTorch: an imperative style, high-performance deep learning library. In: Proceedings of the 33rd International Conference on Neural Information Processing Systems, 2019. 32
58 Krizhevsky A. Learning multiple layers of features from tiny images. 2009. https://cave.cs.toronto.edu/kriz/learning-features-2009-TR.pdf
59 Deng J, Dong W, Socher R, et al. ImageNet: a large-scale hierarchical image database. In: Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition, 2009. 248–255
60 Chen L C, Papandreou G, Kokkinos I, et al. DeepLab: semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs. IEEE Trans Pattern Anal Mach Intell, 2017, 40: 834–848
61 Cordts M, Omran M, Ramos S, et al. The Cityscapes dataset for semantic urban scene understanding. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2016. 3213–3223
62 Zhou B, Zhao H, Puig X, et al. Scene parsing through ade20k dataset. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2017. 633–641
63 Lin T Y, Maire M, Belongie S, et al. Microsoft COCO: common objects in context. In: Proceedings of the European Conference on Computer Vision, 2014. 740–755
64 Ding X, Hao T, Tan J, et al. ResRep: lossless CNN pruning via decoupling remembering and forgetting. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021. 4510–4520
65 Wang H, Qin C, Zhang Y, et al. Neural pruning via growing regularization. ArXiv: 2012.09243
66 Hou Z, Qin M, Sun F, et al. CHEX: channel exploration for cnn model compression. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022. 12287–12298
67 Chin T W, Ding R, Zhang C, et al. Towards efficient model compression via learned global ranking. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020. 1518–1528
68 Guo J, Ouyang W, Xu D. Channel pruning guided by classification loss and feature importance. In: Proceedings of the AAAI Conference on Artificial Intelligence, 2020. 10885–10892
69 Wang Z, Xie H, Wang Y, et al. DCFP: distribution calibrated filter pruning for lightweight and accurate long-tail semantic segmentation. IEEE Trans Circuits Syst Video Technol, 2024, 34: 6063–6076
70 Khaki S, Luo W. CFDP: common frequency domain pruning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2023. 4715–4724
71 Zhong S H, You Z, Zhang J, et al. One less reason for filter pruning: gaining free adversarial robustness with structured grouped kernel pruning. In: Proceedings of the 37th International Conference on Neural Information Processing System, 2023. 62032–62061
72 Zhang M, Wang L, Campos D G C, et al. Weighted mutual learning with diversity-driven model compression. In: Proceedings of the 36th International Conference on Neural Information Processing Systems, 2022
73 Lu X, Dong W, Li X, et al. Adaptive search-and-training for robust and efficient network pruning. IEEE Trans Pattern Anal Mach Intell, 2023, 45: 9325–9338
74 Ruan X, Liu Y, Li B, et al. DPFPS: dynamic and progressive filter pruning for compressing convolutional neural networks from scratch. In: Proceedings of the AAAI Conference on Artificial Intelligence, 2021. 2495–2503
75 Park J H, Kim Y, Kim J, et al. Dynamic structure pruning for compressing CNNs. In: Proceedings of the AAAI Conference on Artificial Intelligence, 2023. 9408–9416
76 He Y, Liu P, Zhu L, et al. Filter pruning by switching to neighboring CNNs with good attributes. IEEE Trans Neural Netw Learn Syst, 2023, 34: 8044–8056
77 Yang L, Gu S, Shen C, et al. Skeleton neural networks via low-rank guided filter pruning. IEEE Trans Circuits Syst Video Technol, 2023, 33: 7197–7211
78 Yuan T, Li Z, Liu B, et al. ARPruning: an automatic channel pruning based on attention map ranking. Neural Netws, 2024, 174: 106220
79 Gao S, Zhang Y, Huang F, et al. Bilevelpruning: unified dynamic and static channel pruning for convolutional neural networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2024. 16090–16100
80 Cheng H, Zhang M, Shi J Q. Influence function based second-order channel pruning: evaluating true loss changes for pruning is possible without retraining. IEEE Trans Pattern Anal Mach Intell, 2024, 46: 9023–9037
81 Lee D, Lee E, Hwang Y. Pruning from scratch via shared pruning module and nuclear norm-based regularization. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, 2024. 1393–1402
82 Wang Y, Guo S, Guo J, et al. Towards performance-maximizing neural network pruning via global channel attention. Neural Netws, 2024, 171: 104–113
83 Xie W, Yuan M, Ma J, et al. Adaptive pruning of channel spatial dependability in convolutional neural networks. In: Proceedings of the

ACM International Conference on Multimedia, 2024. 6073–6082
84 Ding X, Ding G, Han J, et al. Auto-balanced filter pruning for efficient convolutional neural networks. In: Proceedings of the AAAI Conference on Artificial Intelligence, 2018
85 Lin S, Ji R, Yan C, et al. Towards optimal structured CNN pruning via generative adversarial learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019. 2790–2799
86 Park M, Kim D, Park C, et al. REPrune: channel pruning via kernel representative selection. In: Proceedings of the AAAI Conference on Artificial Intelligence, 2024. 14545–14553
87 Ding X, Ding G, Guo Y, et al. Centripetal SGD for pruning very deep convolutional networks with complicated structure. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019. 4943–4953
88 Li Y, van Gemert J C, Hoefler T, et al. Differentiable transportation pruning. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2023. 16957–16967
89 Zhang Y, Freris N M. Adaptive filter pruning via sensitivity feedback. IEEE Trans Neural Netw Learn Syst, 2024, 35: 10996–11008
90 Wang C, Grosse R, Fidler S, et al. EigenDamage: structured pruning in the Kronecker-factored eigenbasis. In: Proceedings of the International Conference on Machine Learning, 2019. 6566–6575
91 Huang Z, Wang N. Data-driven sparse structure selection for deep neural networks. In: Proceedings of the European Conference on Computer Vision, 2018. 304–320
92 Wang H, Fu Y. Trainability preserving neural pruning. In: Proceedings of the International Conference on Learning Representations, 2023
93 Liu Z, Mu H, Zhang X, et al. MetaPruning: meta learning for automatic neural network channel pruning. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019. 3296–3305
94 Peng H, Wu J, Chen S, et al. Collaborative channel pruning for deep networks. In: Proceedings of the International Conference on Machine Learning, 2019. 5113–5122
95 Liu L, Zhang S, Kuang Z, et al. Group Fisher pruning for practical network compression. In: Proceedings of the International Conference on Machine Learning, 2021. 7021–7032
96 Li Y, Gu S, Mayer C, et al. Group sparsity: the hinge between filter pruning and decomposition for network compression. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020. 8018–8027
97 Guan Y, Liu N, Zhao P, et al. DAIS: automatic channel pruning via differentiable annealing indicator search. IEEE Trans Neural Netw Learn Syst, 2023, 34: 9847–9858
98 Lin H, Peng Y F, Bie L, et al. Channel pruning on frequency response. Sci China Inf Sci, 2025, 68: 112102
99 Yu J, Huang T. Autoslim: towards one-shot architecture search for channel numbers. ArXiv: 1903.11728
100 Song Z, Xu Y, He Z, et al. Cp-ViT: cascade vision transformer pruning via progressive sparsity prediction. ArXiv: 2203.04570
101 Guo S, Wang Y, Li Q, et al. DMCP: differentiable Markov channel pruning for neural networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020. 1539–1547