Article:
Li, Chunlei, Li, Huanyu, Gao, Guangshuai et al. (2 more authors) (2023) An accelerating convolutional neural networks via a 2D entropy based-adaptive filter search method for image recognition. APPLIED SOFT COMPUTING. 110326. ISSN: 1568-4946

https://doi.org/10.1016/j.asoc.2023.110326

Applied Soft Computing Journal

An accelerating convolutional neural networks via a 2D entropy based-adaptive filter search method for image recognition

--Manuscript Draft--

Manuscript Number:	ASOC-D-22-04168R1
Article Type:	Full Length Article
Keywords:	CNN acceleration; Image Recognition; Filter search; 2D information entropy
Corresponding Author:	chunlei li Zhongyuan University of Technology CHINA
First Author:	Chunlei li
Order of Authors:	Chunlei li
	Huanyu Li
	Guangshuai Gao
	Zhoufeng Liu
	Pengcheng Liu
Abstract:	The success of CNNs for various vision tasks has been accompanied by a significant increase in required FLOPs and parameter quantities, which has impeded the deployment of CNNs on devices with limited computing resources and power budgets. Network pruning, which compresses and accelerates CNN models, is an effective solution to this issue. Some studies have considered pruning as a special case of neural network search (NAS) in recent years. However, existing techniques are often computationally complex or prone to sub-optimal pruning results. As such, this paper proposes a novel acceleration method via a 2D Entropy based-Adaptive Filter Search (2EAFS). The importance of corresponding filters, measured by utilizing the amount of information contained in feature maps, is employed as a theoretical guide to simplify the complex exhaustive search process. Information entropy is then normalized layer by layer and the resulting value is used to calculate a layer-wise importance score in a single step. Additionally, a sparse constraint equation is constructed based on the negative correlation between filter pruning rates and the importance of convolutional layers. The Nelder-Mead search algorithm is then adopted to quickly and adaptively determine the optimal pruning architecture. Finally, importance weights are inherited using the pruning rate and 2D entropy and model performance are restored through fine-tuning. Extensive experiments conducted with the CIFAR-10/100, ILSVRC-2012, NWPU-RESISC45 and CUB-200-2011 datasets showed this approach achieved considerable accuracy increases, with significant reductions in FLOPs and required parameters that surpassed current state-of-the-art methods by a wide margin. For example, 2EAFS achieved a 44.1% reduction in FLOPs over ResNet-50, with only a 0.53% Top-5 accuracy decrease for ILSVRC-2012.

An accelerating convolutional neural networks via a 2D entropy based-adaptive filter search method for image recognition

Chunlei Li[a], Huanyu Li[b], Guangshuai Gao[c], Zhoufeng Liu[a], Pengcheng Liu[d]

[a]*School of Electrical and Information Engineering, Zhongyuan University of Technology, Zhengzhou, 450007, Henan, China*
[b]*College of Oceanography and Space Informatics, China University of Petroleum (East China), Qingdao, 266580, Shandong, China*
[c]*State Key Laboratory of Virtual Reality Technology and Systems, Beihang Universityzation, Beijing, 100191, China*
[d]*Department of Computer Science, University of York, York, YO10, U.K*

Abstract

The success of CNNs for various vision tasks has been accompanied by a significant increase in required FLOPs and parameter quantities, which has impeded the deployment of CNNs on devices with limited computing resources and power budgets. Network pruning, which compresses and accelerates CNN models, is an effective solution to this issue. Some studies have considered pruning as a special case of neural network search (NAS) in recent years. However, existing techniques are often computationally complex or prone to sub-optimal pruning results. As such, this paper proposes a novel acceleration method via a **2**D **E**ntropy based-**A**daptive **F**ilter **S**earch (2EAFS).The importance of corresponding filters, measured by utilizing the amount of information contained in feature maps, is employed as a theoretical guide to simplify the complex exhaustive search process. Information entropy is then normalized layer by layer and the resulting value is used to calculate a layer-wise importance score in a single step. Additionally, a sparse constraint equation is constructed based on the negative correlation between filter pruning rates and the importance of convolutional layers. The Nelder-Mead search algorithm is then adopted to quickly and adaptively determine the optimal pruning architecture. Finally, importance weights are inherited using the pruning rate and 2D entropy and model performance are restored through fine-tuning. Extensive experiments conducted

with the CIFAR-10/100, ILSVRC-2012, NWPU-RESISC45 and CUB-200-2011 datasets showed this approach achieved considerable accuracy increases, with significant reductions in FLOPs and required parameters that surpassed current state-of-the-art methods by a wide margin. For example, 2EAFS achieved a 44.1% reduction in FLOPs over ResNet-50, with only a 0.53% Top-5 accuracy decrease for ILSVRC-2012.

Keywords: CNN acceleration, Image recognition, Filter search, 2D information entropy

1. Introduction

In recent years, convolutional neural networks (CNNs) have achieved unprecedented results that have been widely applied in various computer vision tasks, such as autonomous driving [1, 2], industrial defect detection [3, 4], and intelligent security [5, 6]. However, as identification and detection performance continue to increase, so does network model capacity (i.e., width and depth). This in turn places higher requirements on the storage space, computing efficiency, and power consumption of computing devices. For instance, ResNet-50 [7] requires 25.26M parameters and 4.11B floating point operations (FLOPs) for forward inference with an image of size 224×224. Considering the cost and practical environment of industrial applications, we generally cannot provide devices with sufficient storage and computing power to meet the needs of developing CNNs, especially on mobile platforms such as phones, micro-robots, and drones.

Various studies have been developed in recent years to address the problem of reducing parameters and FLOPs, such as compact network design [8, 9], low-bit knowledge distillation (KD) [10, 11], and quantization [12, 13]. For instance, ShuffleNetV1 [8] employs group convolutions and channel shuffling to enhance the performance of compact models. Hinton et al. [10] introduced knowledge distillation, which primarily transfers feature information from a complex teacher model to a lightweight student model. Rastegari et al. [12] quantized weights and activations to 1-bit, enabling the model to achieve real-time detection speed and a low resource footprint. However, these methods require specific acceleration frameworks (e.g., quantization) or specialized knowledge for manual design (i.e., compact network design and KD).

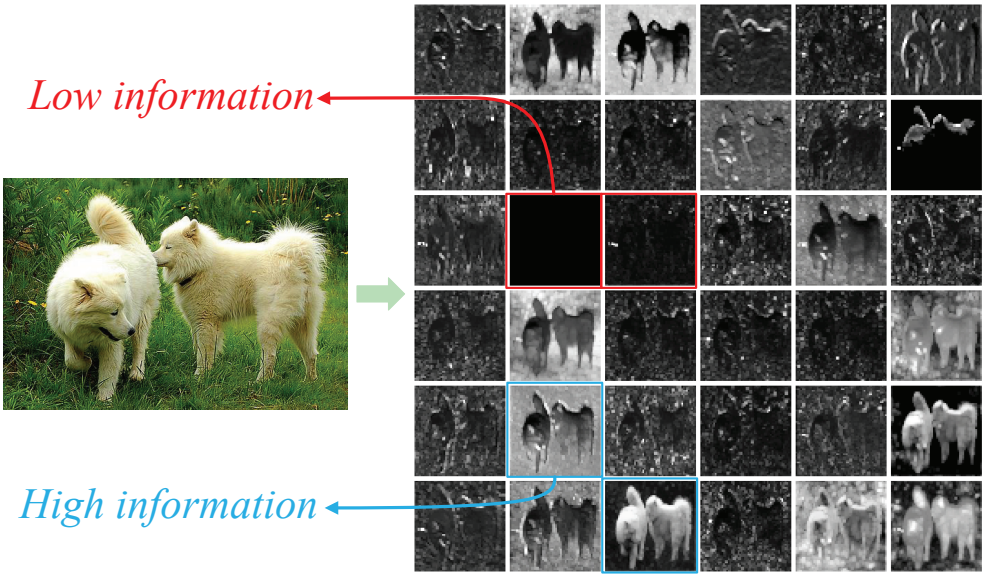


Figure 1: Feature maps generated by the first residual group in ResNet-50. The blue/red boxes indicate high/low information.

Network pruning has achieved considerable success due to its simplicity of operation, implementation efficiency, reduced network complexity, and high resolution for over-fitting problems [14, 15, 16, 17, 18]. Differences in sparse structures can be used to divide pruning methods into unstructured and structured types. Unstructured pruning reduces the computational cost of CNNs by removing individual neurons in filters or connections that contribute less to fully-connected layers. This type of approach requires specific designs for hardware accelerators; otherwise, it is difficult to achieve significant speed-ups. Structured pruning can also directly remove unimportant channels (filters) without changing the network structure, thus allowing effective compression and acceleration using basic linear algebra subprograms (BLAS). The resulting pruned network can be further accelerated by quantization and knowledge distillation. In this paper, we focus on filter pruning, enabling the model to achieve compression and acceleration while preserving performance, providing an effective solution for devices with limited computing power.

Previous studies on structured pruning primarily rely on designed filters and channel importance criteria, without considering the influence of information entropy (contained in feature maps) on model accuracy, which limits

pruning effectiveness. However, rich feature information extracted from well-trained CNNs can provide a comprehensive understanding of an input image. For example, Fig. 1 shows a series of feature maps generated from an image by ResNet-50 during forward inference, in which there are obvious differences in the resulting visualizations of each feature map. For example, some maps (blue boxes) contain more useful feature information (e.g., edges and contours), while others (red boxes) include less feature information. Information entropy can effectively provide quantitative measurements that can be used to represent the richness of information in an image. In this study, information entropy was applied to identify useful feature information and remove redundant features, producing a more efficient neural network compression model.

This paper proposes a simple yet effective **2**D **E**ntropy based-**A**daptive **F**ilter **S**earch (2EAFS) method for fast CNN acceleration, which is illustrated in Fig. 2. The importance of corresponding filters is first measured using the 2D entropy of feature maps, which simplifies the complexity of model pruning without imposing additional sparsity constraints or retraining requirements. FLOPs constraints are then established based on the negative correlation between filter pruning rates and the importance of convolutional layers. The Nelder-Mead search algorithm [19] was used to rapidly and adaptively determine a pruning architecture for each layer, producing a compact sub-network that meets budget constraints. Critical weights are then inherited based on the pruning rate and 2D entropy, while model performance is restored by fine-tuning. This approach can automatically obtain better pruning rates than empirical settings, dramatically simplifying the original structure. This filter weight inheritance based on 2D entropy correspondence was able to significantly outperform random weight inheritance. The contributions of this study can be summarized as follows:

1. A novel network pruning method (i.e., 2EAFS) is proposed, adopting 2D entropy feature evaluation and fast adaptive filter search. This end-to-end pruning framework simplifies complexity without retraining or hyperparameter tuning of the model.
2. An efficient filter importance evaluation criterion is proposed. Specifically, only mini-batches of data are passed to the CNN model to accurately estimate the importance of filters based on the 2D entropy of the feature maps.
3. A negative correlation between filter pruning rate and convolutional

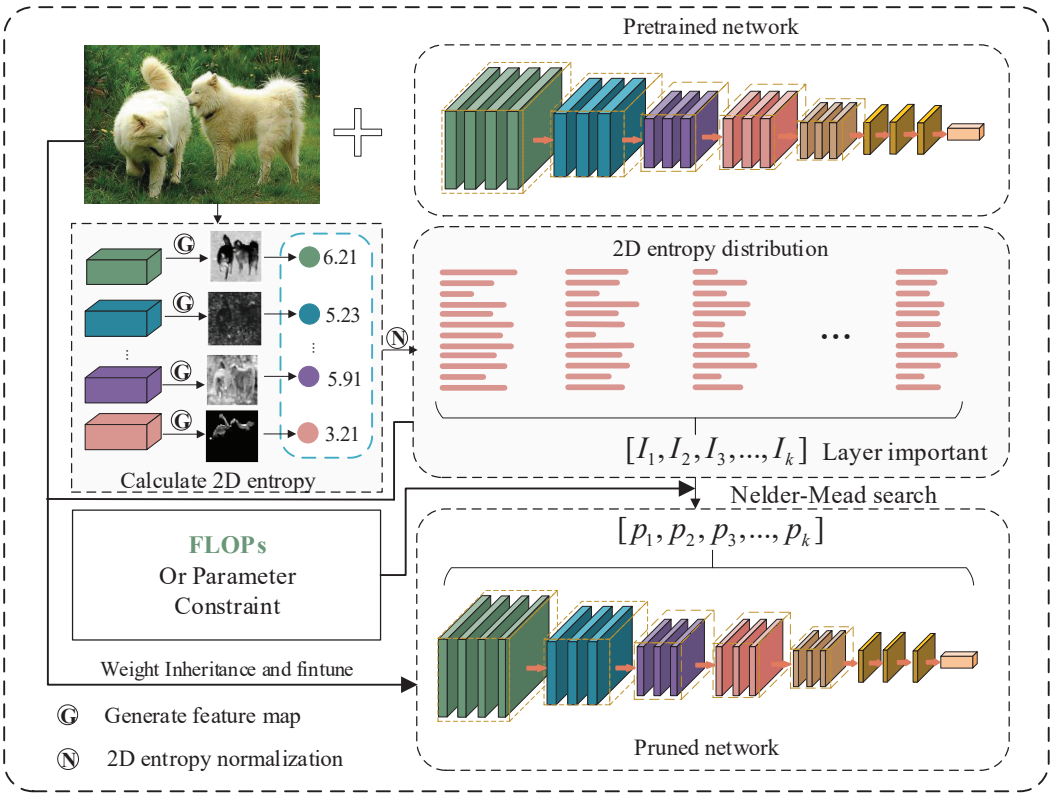


Figure 2: The proposed 2EAFS framework. The importance of corresponding filters was first evaluated using the two-dimensional entropy of the feature map. Constraint equations were then constructed using the importance of convolutional layers, while the sub-network structure was adaptively determined by the Nelder-Mead search algorithm. Finally, important weights were inherited based on the pruning rate and 2D entropy and model performance was restored by fine-tuning.

layer importance is utilized to construct sparse constraint equations. An optimal pruning architecture for each layer is then quickly determined using the Nelder-Mead search algorithm without human involvement.

4. Extensive experiments were conducted with CNNs (e.g., VGG and ResNet) applied to datasets such as CIFAR-10/100, ILSVRC-2012, NWPU-RESISC45 and CUB-200-2011. Results demonstrated the effectiveness and efficiency of 2EAFS in reducing FLOPs and parameter requirements.

The remainder of this paper is organized as follows. Related works on model pruning are discussed in Section II and our proposed 2EAFS framework is then introduced in Section III. Experimental results and corresponding analysis are provided in Section IV. Finally, the paper is concluded in Section V.

2. Related Work

Pruning is highly effective for model compression and can be divided into three categories: unstructured pruning, fine-grained structured sparse, and structured pruning techniques.

2.1. Unstructured pruning

Unstructured pruning reduces the computational costs of a CNN by removing individual neurons in a filter or connections that contribute less to the fully connected layer. The optimal brain damage (OBD) algorithm proposed by LeCun et al. [20], a seminal study in unstructured pruning, uses second-order derivatives to balance training loss and model complexity. A few years later, Srinivas and Babu [21] proposed a data-free technique for pruning fully connected layers to obtain a compact sub-network offering comparable performance. Liu et al. [22] used a 2D discrete cosine transform (DCT) to increase wight sparseness and remove spatial redundancy. Dong et al. [23] proposed a pruning algorithm called L-OBS, which pruned parameters using second-order derivative information from a hierarchical error function, allowing the model to prune large parameter quantities with little degradation in performance. Chen et al. [24] extended the model compression problem from the perspective of constrained Bayesian optimization and introduced a cooling (annealing) solution. However, unstructured pruning produces irregular sparse filters. In addition, a specific hardware acceleration library must be designed according to the required characteristics. Otherwise, it can be difficult to achieve theoretical acceleration effects. 2EFAS can retain the overall structure of the original model and facilitate integration with existing hardware and software structures.

2.2. Fine-grained structured sparse

Fine-grained Structured Sparse usually group weight elements into small dense regions and prune them at the granularity of groups. To achieve better acceleration performance and higher sparsity, researchers have proposed

various grouping methods. Ji et al. [25] proposed a method to rearrange irregular fine-grained sparsity into structured coarse-grained sparsity to bridge the gap between large sparse models and poorer actual acceleration. Lin et al. [26] proposed a novel pattern of 1×N for network pruning that achieves significant CPU acceleration while maintaining high-performance accuracy. Supported by the NVIDIA Ampere Core, N: M sparsity leads to attractive storage and computation efficiency and thus has been extensively studied recently. Zhou et al.[27] proposed the sparse-refined straight-through estimator (SR-STE) and sparse architecture divergence (SAD) to train N: M structured sparse networks, achieving significant acceleration. Zhang et al. [28] proved that N: M learning can be naturally formulated as a combinatorial optimization problem of finding the best combination candidates in a finite collection. Despite the fact that the aforementioned methods can achieve good speed and recognition accuracy on GPUs with Ampere architecture support, the inference speed on CPUs and ARM is still limited by hardware constraints.

2.3. Structured pruning

In contrast, structured pruning directly removes channels and corresponding filters, which is advantageous for forward inferences and can thus be readily deployed in embedded devices. Structured pruning can be roughly divided into three categories (based on the included compact CNN learning process): pretraining-dependency, regularization-retraining, and automatic subnet search pruning.

2.3.1. Pretraining-dependency pruning

This approach prunes a filter based on the intrinsic characteristics of a CNN. The performance of the pruned model can then be enhanced by fine-tuning. For example, Li et al. [14] used an L1-norm to prune filters with low feature information and inherit critical weights. NISP [29] applies feature ranking to measure the importance of each neuron in a final response layer (FRL), which considers the network pruning problem to be a binary integer optimization task. He et al. [30] analyzed the limitations of paradigm-based network pruning and proposed a geometric median-based filter pruning strategy. Lin et al. [31] preserved filter weights with high-rank feature map outputs. Other studies have also considered the effects of parameter pruning on the corresponding loss. For instance, Molchanov et al. [15] proposed evaluating filter importance by considering first-order gradients. Lee et al. [16] directly performed unstructured pruning of randomly initialized weights

using derivatives. Filter pruning has also been implemented using feature reconstruction. He et al. [32] proposed an iterative two-step algorithm to efficiently prune convolutional layers using LASSO regression and linear least-squares. However, the model evaluation functions used in these methods must be specially designed. As a result, they offer low time complexity but exhibit limitations in performance and compression rates. Conversely, 2EFAS utilizes image information entropy to represent the richness of feature maps, thereby solving the problem of insufficient representation of critical information.

2.3.2. Regularization-retraining pruning

Unlike pruning techniques based on the intrinsic properties of neural networks, regularization retraining models introduce sparse constraints and masking strategies to reduce model complexity during the training process. For example, Wen et al. [33] proposed a structured sparsity learning (SSL) method to regularize DNN structures (i.e., filters, channels, filter shapes, and layer depths), thereby achieving hardware-friendly structural sparsity. Liu et al. [34], and Zhao et al. [35] applied regularization to scale factors in batch normalization layers as part of a loss function and culled channels corresponding to lower factors (based on the pruning rate). Huang and Wang [36], and Lin et al. [37] introduced a mask to learn sparse model structures and remove filters corresponding to zero-scale factors. Similarly, SWP [38] is a filter pruning method that uses a pruning skeleton to learn optimal filter shapes effectively. Although these strategies are straightforward and eliminate reliance on pre-training models, the additional constraints require training from scratch, which reduces the generalizability and flexibility of training loss, 2EFAS does not necessitate any modification of the loss function or retraining, but rather only entails fine-tuning, resulting in substantial savings in resources and computation time, making it more conducive for industrial application.

2.3.3. Automatic subnet search

Some recent studies have focused on searching for an optimal structure (i.e., the filter or channel number for each layer) rather than evaluating filter importance. MetaPruning [39] randomly samples each layer's output channel number as input and train a PruningNet to generate high-quality weights for sub-networks of different architectures. An evolution algorithm is then used to search for optimal sub-networks that satisfy these constraints. AMC [40]

accepts the compression rate and accuracy as feedback and uses reinforcement learning to develop an intelligent agent yielding the pruning rate for each layer. However, reinforcement learning typically exhibits unstable convergence, which requires significant effort for parameter tuning. ABCPruner [41] uses an artificial bee colony algorithm to automatically search for efficient network architectures and fine-tune the result to identify the most efficient sub-networks. However, it also requires retraining the subnetwork used for performance evaluations at high computational costs. DMCP [42] models channel pruning as a differentiable Markov process, thus, eliminating the burden of searching multiple architectures. Although this approach can reduce human intervention, it requires training the network from scratch with different parameter settings, which is time-consuming and requires excessive energy consumption, 2EFAS can directly obtain the optimal pruned network under sparse constraints. This also avoids the difficulty of restructuring the search when the constraint targets or datasets change, and the process is simple and intuitive.

3. Proposed Method

This section details the proposed approach illustrated in Fig. 2. The filter pruning problem is first revisited, and the proposed filter importance evaluation criteria are then introduced. This metric is based on 2D information entropy in feature maps and allows for fast determination of the optimal filter pruning rate. The pruning framework and implementation details are also introduced. Pruning strategies based on multi-branch networks (i.e., ResNet [7]) are then discussed.

3.1. Preliminary

The primary notations used in this paper are summarized in Table 1, and the associated neural network pruning problem can be described as follows. Given a training set $\mathcal{X}_{\text{train}}$ and testing set $\mathcal{T}_{\text{test}}$, we attempt to determine the optimal pruning rate p_k and weights $\mathbf{w}'_k$ for each layer, achieving the highest accuracy while satisfying a set of sparse constraints. To this end, the pruning problem can be expressed as:

$$\begin{aligned} &(F')^* = \arg\max_{F'} \operatorname{Acc}\left(\mathcal{M}_p\left(F', \mathbf{W}'; \mathcal{X}_{\text{train}}\right); \mathcal{X}_{\text{test}}\right) \\ &\text{s.t. } \mathcal{F}\left(\mathcal{N}_p\left(C', \mathbf{W}'; \mathcal{X}_{\text{train}}\right)\right) < \mathcal{C} \end{aligned} \tag{1}$$

Table 1: NOTATIONS AND THEIR ASSOCIATED MEANINGS.

Notation	Meaning
$\mathcal{M}(\cdot)$ and $\mathcal{M}_p(\cdot)$	Pretrained and pruned network model
$\mathcal{X}_{\text{train}}$ and $\mathcal{T}_{\text{test}}$	Training set and testing set
$\mathbf{W} = \{\mathbf{w}_k\}_{k=1}^{K} \in \mathbb{R}^{f_k \times f_{k-1} \times h_k \times w_k}$	$\mathbf{w}_k$ are the filters in the k^{th} layer
$\mathbf{W}' = \{\mathbf{w}'_k\}_{k=1}^{K} \in \mathbb{R}^{f'_k \times f'_{k-1} \times h_k \times w_k}$	$\mathbf{w}'_k$ are the filters of the k^{th} layer after pruning
p_k	The pruning rate for the k^{th} layer
f_k and f'_k	The number of filters in the k^{th} layer before and after pruning
h_k and w_k	The height and width of filters in the k^{th} layer
$F = \{f_1, f_2, \cdots, f_K\}$ and $F' = \{f'_1, f'_2, \cdots, f'_K\}$	CNN network structure before and after pruning
$\mathbf{O} = \{\mathbf{o}_k\}_{k=1}^{K} \in \mathbb{R}^{f_k \times m_k \times n_k}$	The feature map generated after convolution operations: batch normalization, and activation.
m_k and n_k	The height and width of the output feature map in the k^{th} layer.

where $\text{Acc}(\cdot)$ denotes the accuracy of $\mathcal{T}_{\text{test}}$ applied to $\mathcal{M}_p(\cdot)$ with a structure F'. It should be noted the parameters of the pruned network can provide a weight inheritance for the original network or random initialization parameters. $\mathcal{F}(\cdot)$ is the set of sparse constraint limits, such as FLOPs, parameters, power, and latency, which require identifying an optimal trade-off between accuracy and constraints.

3.2. Information measurement based on 2D entropy

Previous studies [43] have used feature map sparsity to remove filters. However, this data-driven approach depends heavily on the distribution of input images. It also requires large sample quantities to make reasonable predictions, which involves a complex and time-consuming training process. Image entropy can be used for quantitative information measurements and can also represent the richness of feature maps. The distribution of entropy is also robust across the entire data set, which has been demonstrated through extensive empirical validation. Among the requirements discussed above,

quantization of the output feature map $\mathbf{o}_k^i$ in the i^{th} filter of the k^{th} layer can be defined as follows:

$$\tilde{\mathbf{o}}_k^i = \text{round}\left(\sigma\left(\mathbf{o}_k^i\right) \cdot 255\right) \tag{2}$$

where $\sigma(\cdot)$ is the sigmoid activation function used to normalize a feature map and round$(\cdot)$ is a rounding function used to generate 8-bit grayscale feature maps.

Calculations based on Shannon's entropy theory can only represent entropy values for a whole image. As such, they cannot represent the spatial characteristics of image pixel distributions, which affect the accuracy of information evaluation [44]. 2D information entropy is introduced to represent spatial distribution characteristics in the feature maps to address this issue. Specifically, we selected grey values adjacent to the image mean to describe spatial feature quantities for grey distributions and form a feature binary with image pixels. These grey values can be denoted as (i, j), where i represents a grey pixel value $(0 \leq i \leq 255)$, and j is the neighborhood grey mean. The term $q_{ij} = f(i, j)/S^2$ denotes the comprehensive characteristics of grey values at individual pixel positions and grey distributions for surrounding pixels. The term $f(i, j)$ is the occurrence frequency for the feature binary (i, j) and S is the image scale. The discrete 2D image entropy can then be expressed as:

$$H = -\sum_{i,j=0}^{255} q_{ij} \log_2 q_{ij} \tag{3}$$

Experiments have shown the average 2D entropy distribution for feature maps generated from a few samples in the same filter remains mostly unchanged compared to those of larger data quantities. As such, only small sub-batches of data must be passed to the CNN model to accurately estimate the importance of filters based on the 2D entropy of feature maps, thus saving significant forward inference overhead. Based on this approach, the expected 2D image entropy in each feature map can be defined as:

$$\bar{H}_k^i = \frac{\sum_{g=1}^{G} H_k^i(g)}{G} \tag{4}$$

where G is the number of image samples and $H_k^i(g)$ is the 2D feature map entropy for the i^{th} filter at the k^{th} layer in the forward inference. However,

since the range of entropy values is closely related to the size of feature maps, entropy values in different convolutional layers vary significantly. To this end, we propose a max-min normalization to normalize the 2D entropy of each layer to [0, 1] as follows:

$$\bar{H}_k^a \leq \tilde{H}_k^i = \frac{\bar{H}_k^i - \bar{H}_k^a}{\bar{H}_k^b - \bar{H}_k^a} \leq \bar{H}_k^b, \quad a, b \in [1, f_k] \text{ and } a \neq b \tag{5}$$

where $\bar{H}_k^a$ and $\bar{H}_k^b$ are the maximum and minimum values of the feature map entropy in the k^{th} layer, respectively, and a and b are the corresponding filter indices. Thus, we can establish filter importance evaluation criteria based on the 2D information entropy in a feature map, pruning the model at the given pruning rate as follows:

$$\text{Ind}_k = \text{sort}_{p_k}\left(\text{imp}\left(\tilde{H}_k^i\right)\right) \tag{6}$$

where $\text{imp}(\cdot)$, used to measure the importance of corresponding filters in a feature map, is positively related to the magnitude of the entropy values. The term $\text{sort}_{p_k}(\cdot)$ is a descending order function that returns the index Ind_k of a filter to be retained, based on the pruning rate p_k. However, it can be difficult to quickly determine the optimal rate for each layer while satisfying the given constraints, as discussed in the following section.

3.3. Adaptive Filter Search

Several recent studies [39, 41, 42] have modeled neural network pruning as an optimal structure search problem, used to determine the pruning rate in each layer with various intelligent algorithms. However, these techniques exhibit several problems, including the following. (1) The introduction of the pruning rate step size vector reduces the search space, which often results in sub-optimal performance. (2) The structured search must be repeated when budget requirements and datasets are modified, which requires significant computational resources and runtime. (3) The minimum pruning rate in each layer is set based on subjective experience and is not universal. (4) The use of hyper-parameters often leads to issues with tuning optimization, which further increases the complexity of the problem.

The last layer of a CNN tends to provide the best discrimination information and is commonly used for various application tasks. However, intermediate convolutional layers contain far more critical information in practice.

Thus, by analyzing the correlation between layers, a more appropriate pruning method can be developed to achieve higher recognition accuracy while reducing the number of required computations. In this study, we automatically determined the pruning rate for each layer using the importance I_k of individual convolutional layers [45]:

$$I_k = \frac{M_k}{\sum_{k=1}^{K} M_k} \tag{7}$$

where $M_k = \frac{1}{f_k} \sum_{i=1}^{f_k} \tilde{H}_k^i$. After obtaining the importance assessment metrics for each block, we must determine how to map this importance to the actual pruning rates in each convolutional layer. As in previous studies [46], we assumed the filter pruning rate in each layer is negatively correlated with the importance. A pruning rate evaluation function can then be constructed from this assumption:

$$p_k = 1 - \left(\alpha I_k + \beta I_k^2\right) k = 1, 2, 3, \ldots, K \tag{8}$$

where α is used to describe the slope of the first-order linearity and β is the bias correction term introduced to identify the optimal filter distribution while satisfying sparse constraints (FLOPs) used to analyze speedup performance. Eq. (1) converts the generic pruning problem into an attempt to solve for α and β as follows:

$$\begin{aligned} &\left|\mathcal{F}\left(\mathcal{N}_p\left(\left(\alpha I_k + \beta I_k^2\right) * F, \mathbf{W}'; \mathcal{X}_{\text{train}}\right)\right) - \mathcal{C}\right| < \mathcal{T} \\ &\quad \text{s.t. } 0 < \alpha I_k + \beta I_k^2 \leq 1 \quad k = 1, 2, 3, \ldots, K \end{aligned} \tag{9}$$

where $0 < \alpha I_k + \beta I_k^2 \leq 1$ and $\mathcal{T}$ and $\mathcal{C}$ are the tolerance and target computation, respectively. Since $p_k \in (0, 1]$, a range constraint is also involved. The linearity of this relationship allows the problem to be transformed as follows:

$$\begin{aligned} &\left|\mathcal{F}\left(\mathcal{N}_p\left(\left(\alpha I_k + \beta I_k^2\right) * F, \mathbf{W}'; \mathcal{X}_{\text{train}}\right)\right) - \mathcal{C}\right| < \mathcal{T} \\ &\text{s.t. } \max\left(\alpha I_k + \beta I_k^2\right) \leq 1 \\ &\min\left(\alpha I_k + \beta I_k^2\right) > 0 \quad k = 1, 2, 3, \ldots, K \end{aligned} \tag{10}$$

Since the FLOPs constraint is closely related to the number of filters in the front-end and back-end layers, it can be difficult to solve for the values of α and β directly. The Nelder-Mead search algorithm [19] can be used to search the minimum of a multivariate function and does not require a derivable

expression. It also converges to a minimum relatively quickly, saving significant runtime compared to pruning algorithms such as ABCPruner [41]. As such, this paper applied the encapsulated Nelder-Mead search algorithm in scikit-learn under a fixed search domain. Fast determination of the optimal pruning rate for each model layer was achieved as a result. It is worth noting that, although previous studies have demonstrated pruned sub-networks can achieve comparable performance without inheriting original network weights (random initialization), a recent study [47] proved the inheritance of important weights in a pruned network could help a sub-network converge to higher accuracy without investing more training resources.

3.4. Pruning Implementation and Multi-Branch Network Processing

The proposed pruning process (2EAFS), based on the above analysis, is shown in Algorithm 1. The 2D entropy of the output feature map for the layer to be pruned was first calculated using Eq. (3). The corresponding entropy value was then restricted to [0,1] using max-min normalization. The importance of each convolutional layer was then determined using Eq. (7) and a function based on the FLOPs constraint was constructed to adaptively and rapidly determine the ideal pruning rate for each layer. Critical weights were then inherited based on high or low 2D information entropy. Finally, the pruned model was fine-tuned to maintain comparable performance with the pre-trained model.

Common CNN models include efficient plain networks, multi-branch network structures offering higher recognition accuracy, and associated variations. For example, VGG is a typical front-end representation of this structure type [43]. Other network structures have also been developed recently, including ResNet[7]. Due to the singular structure of the VGG network, critical weights can be directly obtained from a 2D entropy evaluation and the Nelder-Mead search algorithm, as shown in Fig. 3(a). In the case of ResNet, this approach involves some limitations due to the introduction of residual connections. For instance, suppose a residual block must be used to complete a sum operation. In this case, it is necessary to ensure the number of input and output channels are consistent, making it difficult to directly prune the last convolutional layer in each residual block. For example, ResNet-56/110 (shown in Fig. 3(b)) includes two 1×1 convolutional layers per residual block. However, the parameters and convolution computations are primarily concentrated in the first layer. In contrast, each residual block in ResNet-50 (shown in Fig. 3(c)) includes two 1×1 and one 3×3 convolutional layer. As

such, most of the calculations are performed in the first two layers and only the last layer is reserved for each residual block of the ResNet, while other layers are pruned.

Algorithm 1 The pruning algorithm based on 2EAFS

Input: The pre-trained model weight is given by $\{\mathbf{w}_k\}_{k=1}^{K}$, the number of fine-tuned iterations is N, and the target FLOPs quantity is $\mathcal{C}$.

Output: The pruned model $\mathcal{M}_p(\cdot)$ and the weight parameters $\{\mathbf{w}'_k\}_{k=1}^{K}$ after fine-tuning.

for all $1 \rightarrow K$ **do**
    2D entropy $H \leftarrow$ Eq.(3)
    $\tilde{H}_k^i \leftarrow$ Eq.(4) and Eq.(5)
end for
$I_k \leftarrow$ Eq.(7)
Construct a $\mathcal{C}$ constrained Eq.(8) and Eq.(9)
$p_k \leftarrow$ Eq.(10) by the Nelder-Mead algorithm
$\{\mathrm{Ind}_k\}_{k=1}^{K} \leftarrow$ Eq.(6)
Inherit important weights $\{\mathbf{w}'_k\}_{k=1}^{K}$ by $\{\mathrm{Ind}_k\}_{k=1}^{K}$
while $1 \rightarrow N$ **do**
    Fine-tune the pruned model $\mathcal{M}_p(\cdot)$
end while

4. Experiments

In this section, we demonstrate the compression performance of the proposed method applied to the CIFAR10/100 and ILSVRC2012 datasets using different network architectures. Implementation details for these experiments are first explained and comparisons are then provided with some popular techniques. Finally, the causes of this improved performance displayed by the pruned model are discussed.

4.1. Experimental Details

4.1.1. Benchmark Datasets and Models

Five benchmark datasets, CIFAR-10 [48], CIFAR-100 [48], ILSVRC-2012 [49], NWPU-RESISC45[50] and CUB-200-2011[51] were used to conduct a series of validation experiments. To facilitate a comparison with other methods, the performance of the proposed pruning technique was investigated using mainstream CNN models, including plain networks (i.e., VGG-16 [52])

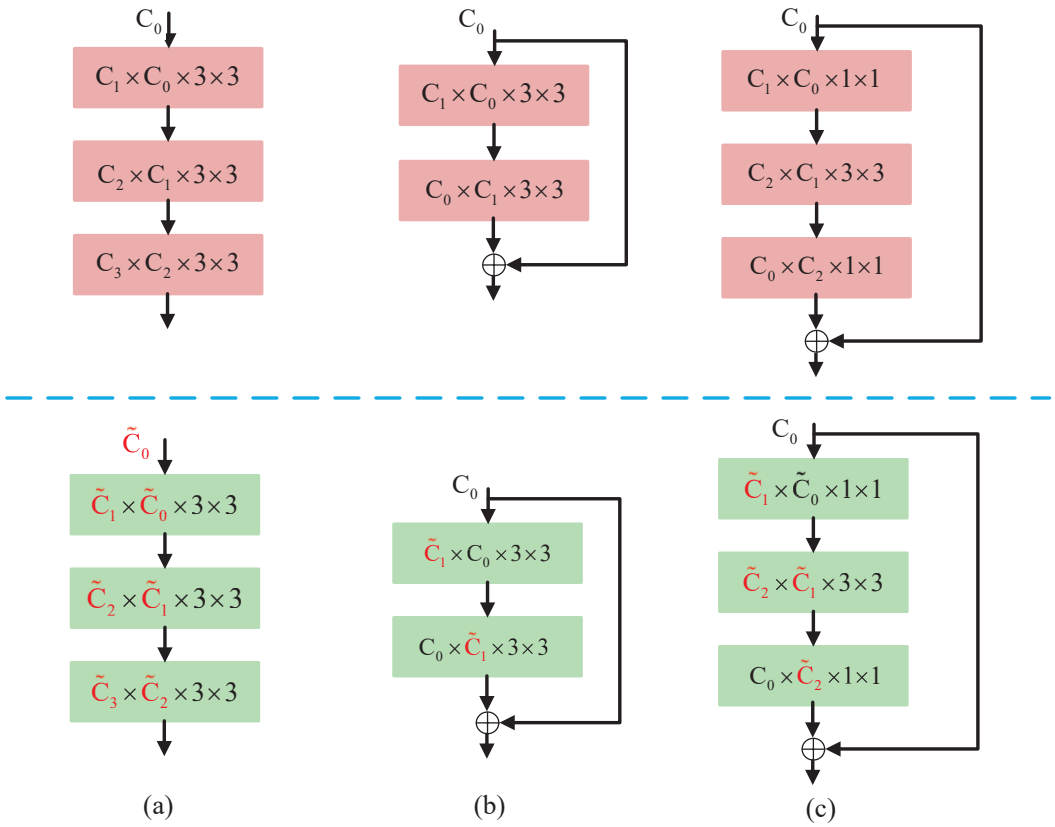


Figure 3: Pruning strategies for the (a) regular structure, (b) residual block1, and (c) residual block2 in the proposed 2EAFS. The online model represents the unpruned structure, while offline denotes the pruned structure.

and multi-branch networks (i.e., ResNet). CIFIR-10 and CIFIR-100 are image datasets with sample sizes of 32x32, for which 50k images were used for training and 10k were used for testing. The primary difference is the former includes only 10 categories while the latter spans 100. ILSVRC-2012 is a large-scale dataset containing 1.28 million training images and 50k validation images from 1,000 categories. NWPU-RESISC45 dataset is a publicly available benchmark for remote sensing image scene classification which contains 31,500 images, covering 45 scene classes with 700 images in each class, we are divided into training sets and test sets according to the 7: 3 ratio. The CUB-200–2011 is the most widely used benchmark for fine-grained image classification. The dataset covers 200 species of birds, including 5,994

training images and 5,794 test images. Specifically, VGG-16, ResNet-56, and ResNet-110, were pruned using CIFAR-10/100, ResNet-50 was pruned on ILSVRC-2012 and NWPU-RESISC45, while VGG-16 (input image size is 224×224) was pruned on CUB-200-2011.

4.1.2. Training Settings

All experiments were optimized using a stochastic gradient descent algorithm (SGD) with a momentum of 0.9. To alleviate overfitting and speed up convergence, CIFAR-10 used a weight decay of 0.005, CIFAR-100 used weight decay values of 0.005 and 0.0005 for VGG-16 and ResNet-56/110, respectively, and ResNet-50 with ILSVRC-2012 and NWPU-RESISC45 used a weight decay of 0.0001. VGG-16, ResNet-56/110, and ResNet-50 included batch sizes of 256, training epochs of 150, 300, and 90 (NWPU-RESISC45 is 30), and initial learning rates of 0.01, 0.01, and 0.1(NWPU-RESISC45 is 0.01), respectively. The learning rates for all CIFAR-10/100 networks were divided by 10 at the 150th and 225th epochs, excluding VGG-16, which divided the learning rate by 10 at the 50th and 100th epochs. In the case of ILSVRC-2012, the learning rate was divided by 10 in the 30th and 60th epochs, and the learning rate for the first 5 epochs was gradually increased to speed up convergence. The training parameters utilized for VGG-16 on CUB-200-2011 were identical to those for CIFAR-10/100. A total of 2,000 images were randomly sampled from each of the benchmarks and architectures to estimate the average 2D entropy of each feature map.

All models were implemented and trained on an NVIDIA Tesla V100 GPU using Pytorch. Random cropping and horizontal flipping were applied in all experiments to augment input images. No other processing steps (e.g., label smoothing, image augmentation, architecture modification, or cosine/linear decay learning rates) were included, since these experiments are specifically intended to demonstrate the performance of the pruning algorithm.

4.1.3. Performance Metrics

Parameter quantities and FLOPs were used to evaluate pruning model memory footprints and acceleration effects, which are mainstream evaluation criteria used in model compression. Corresponding pruning rates (PR) are reported below. Specifically, the Top-1 accuracy is reported for CIFAR-10/100 and both Top-1 and Top-5 accuracies are reported for ILSVRC-2012. Accuracy correction is another vital evaluation metric for fair comparisons, as discussed below.

4.2. Experiments on CIFAR-10

VGG-16. The performance of our proposed filter pruning method was compared with several conventional algorithms, including those based on the intrinsic properties of neural networks, such as L1-Normal [14], Hrank [30], CPMC [53], SSIM-QSFM [54], several regularization-retraining pruning methods such as SSS [36], Hinge [55], VCNNP [35], SWP [38] and GAL [37], and weight sharing method (i,e, SCWC[56]), as shown in Table 2. Compared with state-of-the-art methods (i.e., Hrank [30], CPMC [53], and QSFM-SSIM [54]), the proposed model exhibited lower performance degradation (0.38% vs 1.62% by Hrank [30], 0.28% by CPMC [53], and 1.79% by QSFM-SSIM [54]), larger parameter reductions (74.9% vs. 65.3% by Hrank, 66.0% by CPMC, and 75.0% by QSFM-SSIM), and larger FLOPs reductions (86.1% vs. 82.1% by Hrank, 92.9% by CPMC, and 75.0% by QSFM-SSIM), all representing significant decreases. Both Hrank and 2EAFS failed to achieve better experimental results in highly compressed cases, which we attribute to manually set pruning rates. 2EAFS also produced better experimental results at different pruning rates in the case of regularization-retraining. For example, SSS [37], Hinge [55], VCNNP [35], and GAL-0.05 [37] produced varying degrees of performance degradation (0.69%/0.94% by SSS [37], 0.43% by Hinge [55], 0.07% by VCNNP [35], and 0.19% by GAL-0.05 [37]) when the number of FLOPs was reduced by less than 50% and the parameters were reduced by no more than 80%. In contrast, the proposed method achieved a 65.0% reduction in FLOPs and an 80.3% reduction in parameters, with only a 0.07% performance decrease. Finally, compared with the weight sharing method SCWC, 2EAFS has better performance with similar FLOPs. These results suggest our proposed pruning method offers excellent performance with a simple model structure.

ResNet-56. Pruning results for ResNet-56 applied to CIFAR-10 are shown in Table 3, where it is evident that 2EAFS offers superior performance (-0.54%/1.59% vs. -0.26%/2.54% by Hrank [30] and -0.39%/2.06% by FilterSketch [57] in Top-1 accuracy reduction), with similar FLOP quantities (30.4%/74.7% vs. 29.3%/74.4% by Hrank and 30.4%/74.4% by FilterSketch). 2EAFS was also compared with two property importance-based methods: L1_Normal [14] and NISP [29]. Notable performance improvements (-0.13% vs. -0.02% by L1_Normal and 0.03% by NISP) were also achieved in the case of more significant reductions in FLOPs and parameters. This demonstrates that 2EAFS is adequate for tasks with high compression rates, enabling deployment on resource-constrained devices. Similar observations

Table 2: Performance comparisons for VGG-16 applied to CIFAR-10.

Method	FLOPs↓	Parameters↓	Top-1%	±Acc(%)
L1-Normal[14]	34.3%	64.0%	93.25→93.40	+0.15
SSS[36]	36.3%	66.7%	93.96→93.27	-0.69
Hinge[55]	39.1%	80.1%	94.20→93.59	-0.43
VCNNP[35]	39.1%	73.3%	93.25→93.18	-0.07
GAL-0.05[37]	39.6%	77.6%	93.96→93.77	-0.19
SSS[36]	41.6%	64.0%	93.96→93.02	-0.94
SCWC[56]	41.9%	39.5%	N/A	+0.07
2EAFS	50.0%	71.4%	93.55→93.92	+0.37
Hrank[30]	53.5%	82.9%	93.96→93.43	-0.53
2EAFS	65.0%	80.3%	93.55→93.62	+0.07
Hrank[30]	65.3%	82.1%	93.96→92.34	-1.62
CPMC[53]	66.0%	92.9%	93.68→93.40	-0.28
SCWC[56]	70.3%	69.4%	N/A	-0.67
SWP[38]	71.2%	92.7%	93.25→92.85	-0.40
QSFM-SSIM[54]	75.0%	75.0%	93.39→91.60	-1.79
2EAFS	74.9%	86.1%	93.55→93.17	-0.38

of the adaptive importance-based methods (GAL-0.6 [37] and SCP [58]) suggested that 2EAFS achieved better parameter compression (53.7% vs. 11.8% for GAL-0.6 and 48.4% for SCP) without performance degradation (-0.13% vs. 0.28% by GAL-0.6 and 0.46% by SCP).

ResNet-110. The performance of ResNet-110 applied to CIFAR-10 was also analyzed, as shown in Table 4. Similar to the results produced by ResNet-56, 2EAFS achieved better accuracy (94.40% vs. 93.61%) compared to the baseline model, with a 43.1% reduction in FLOPs and a 43.0% reduction in parameters. Compared with FilterSketch [57] and Hrank [30], 2EAFS again achieved lower performance degradation (0.02% vs. 0.06% and 0.85%, respectively), with a slight reduction in complexity. This model also outperformed NISP [29] and GAL-0.5 [37], achieving better recognition performance (93.59% vs. 93.38% and 92.55%, respectively). 2EAFS further accelerated the required calculations (70.6% vs. 43.8% and 48.5%) and reduced the overload (71.1% vs. 43.3% and 44.8%), indicating the proposed technique to be highly simple and efficient for multi-branch networks, especially residual networks. Fig. 4 shows the resulting performance degradation of the proposed pruning method applied to ResNet-56 and ResNet-110 under

Table 3: Performance comparisons for Resnet-56 applied to CIFAR-10.

Method	FLOPs↓	Parameters↓	Top-1%	±Acc(%)
DeepPruningES[59]	21.3%	N/A	93.37→91.89	-1.48
L1-Normal[14]	27.6%	14.1%	93.06→93.04	-0.02
Hrank[30]	29.3%	16.8%	93.26→93.52	+0.26
FilterSketch[57]	30.4%	20.6%	93.26→93.65	+0.39
2EAFS	30.4%	34.1%	93.26→93.80	+0.54
SCWC[56]	32.6%	33.2%	N/A	+0.10
GAL-0.6[37]	37.6%	11.8%	92.98→93.26	-0.28
FilterSketch[57]	41.5%	41.2%	93.26→93.19	-0.07
SCWC[56]	41.8%	43.0%	N/A	-0.04
NIPS[29]	43.6%	43.6%	93.06→93.01	-0.03
CP[32]	50.0%	N/A	92.80→91.80	-1.00
AMC[40]	50.0%	N/A	91.80→91.90	-0.90
DCP[60]	50.0%	N/A	93.80→93.59	-0.21
DMC[61]	50.0%	N/A	93.62→93.69	+0.07
2EAFS	50.7%	53.7%	93.26→93.39	+0.13
SCP[58]	51.5%	48.4%	93.69→93.23	-0.46
SFP[62]	52.6%	N/A	93.59→93.35	-0.24
LFPC[63]	52.9%	N/A	93.59→93.34	-0.25
Aakash et al.[64]	53.9%	51.3%	94.11→93.37	-0.74
Hrank[30]	74.4%	68.1%	93.26→90.72	-2.54
FilterSketch[57]	74.4%	71.8%	93.26→91.20	-2.06
2EAFS	74.7%	76.7%	93.26→91.67	-1.59

different FLOPs conditions. Several state-of-the-art methods were compared to demonstrate the advantages of 2EAFS. It is evident from the figure that model recognition effects decreased at varying rates as the pruning rate increased. However, 2EAFS consistently achieved better Top-1 accuracy compared to other pruning models with similar computational effort, indicating the proposed method can achieve maximum compression and acceleration while maintaining accuracy.

4.3. Experiments on CIFAR-100

Experiments were also conducted using three networks applied to the more complex CIFAR-100 dataset. Seven model compression methods were compared, including VCNNP [35], CPGMI [58], CPMC [53], G-pruning [68],

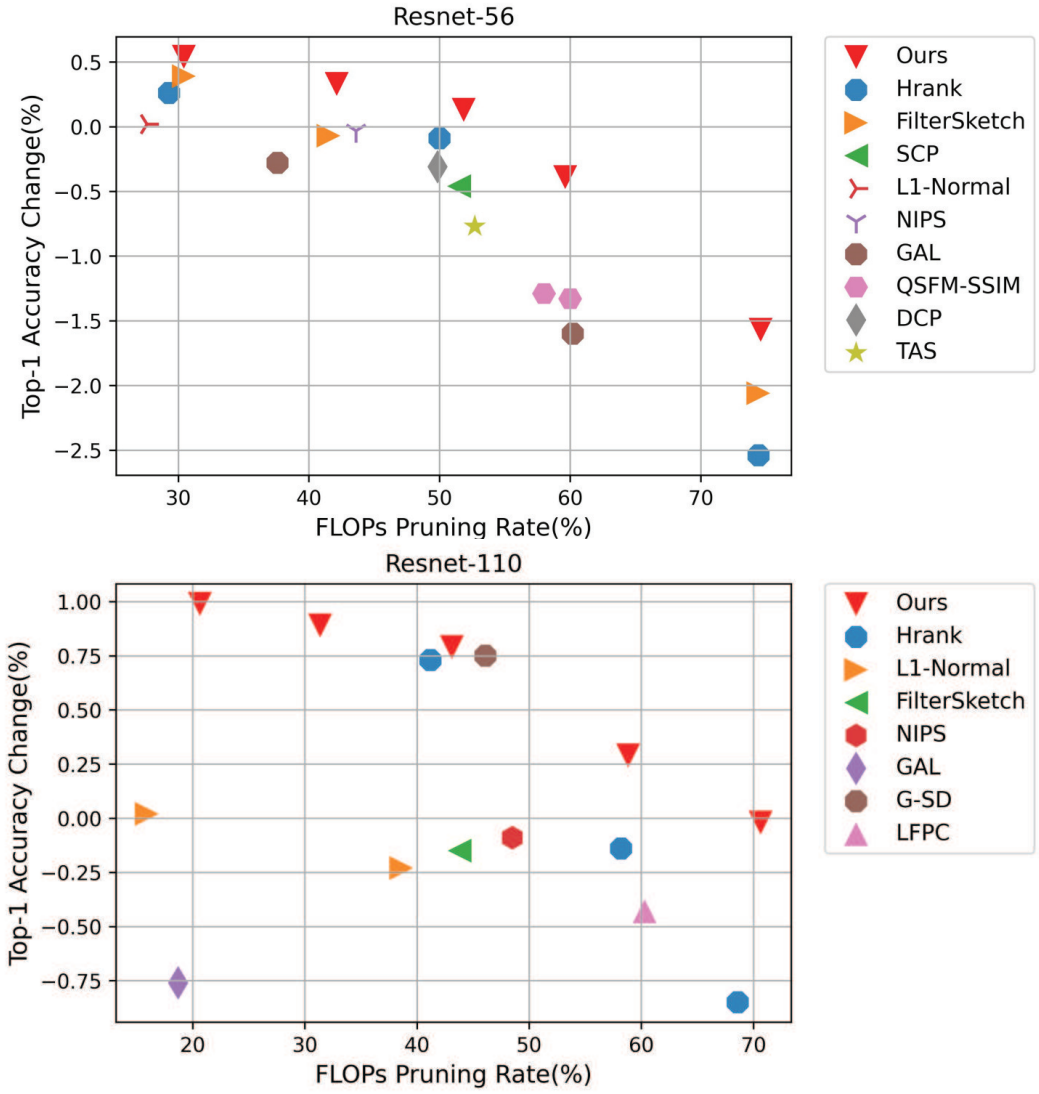


Figure 4: Comparisons of performance degradation for existing methods and 2EAFS under varying FLOPs quantities. Experiments were performed using the CIFAR-10 dataset.

MIL [69], BNP [70], and SFP [62], as shown in Table 5. Specifically, in the case of the simple VGG-16 model structure, the proposed method achieved better time and spatial complexity than VCNNP [35] and CPGMI [58] with comparable performance (73.64% vs. 73.33% and 73.53%). In addition, 2EAFS achieved outstanding performance in terms of accuracy degradation

Table 4: Performance comparisons for ResNet-110 applied to CIFAR-10.

Method	FLOPs↓	Parameters↓	Top-1%	±Acc(%)
SFP[62]	14.6%	N/A	93.68→93.83	+0.15
L1-Normal[14]	15.9%	2.3	93.53→93.55	+0.02
Liu et al.[65]	15.9%	2.3	93.14→93.22	+0.08
L1-Normal[14]	38.6%	32.4%	93.53→93.30	-0.23
SFP[62]	40.8%	N/A	93.68→93.86	+0.18
CNN-FCF[66]	43.1%	43.2%	93.58→93.67	+0.09
Hrank[30]	41.2%	44.8%	93.50→94.23	+0.73
2EAFS	43.1%	43.0%	93.61→94.40	+0.79
NISP[29]	43.8%	43.3%	93.53→93.38	-0.15
GAL-0.5[37]	48.5%	44.8%	93.50→92.55	-0.95
Hrank[30]	58.2%	59.2%	93.50→93.36	-0.14
2EAFS	58.8%	58.6%	93.61→93.90	+0.29
FilterSketch[57]	63.3%	59.9%	93.50→93.44	-0.06
Hrank[30]	68.6%	68.7%	93.50→92.65	-0.85
DeepPruningES[59]	64.8%	N/A	93.80→91.34	-2.46
2EAFS	70.6%	71.1%	93.61→93.59	-0.02
CNN-FCF[66]	70.8%	69.5%	93.58→92.96	-0.62
Nima et.al[67]	N/A	78.0%	94.27→93.00	-1.27

when compressing ResNet-56. For example, our approach produced lower performance degradation under the condition of halving model computations (1.87% vs. 2.86% for BNP and 2.61% for SFP). To further demonstrate the effectiveness of our algorithm, acceleration effects (37.3%, 45.1%, and 53.0%) and the performance (72.85%, 72.30%, and 71.89%) of ResNet-110 under different parameter compression ratios (44.1%, 51.2%, and 58.2%) are reported in Table 5. The results of these experiments demonstrate the proposed data-driven methodology, 2EAFS, can achieve better acceleration effects in high compression states.

4.4. Experiments on ILSVRC-2012

To further demonstrate the effectiveness of our approach for more complex datasets, we compared 2EAFS with several advanced algorithms, including those based on the intrinsic properties of neural networks i.e.,CP [32], Hrank [30], [71] and [74] ,automatic subnet search such as ABCpruner [41] and PSO[73], and some regularization-retraining pruning methods (i.e., SSS

Table 5: Performance comparisons for VGG-16 and ResNet-56/110 applied to CIFAR-100.
PR is used to express the reduction rate for parameters or FLOPs.

Model	Method	FLOPs	PR	Parameters	PR	Top-1%	±Acc(%)
VGG-16	VCNNP[35]	256.00M	18.1%	9.14M	37.9%	73.26→73.33	+0.07
	CPGMI[58]	198.20M	37.1%	4.99M	37.1%	73.26→73.53	+0.27
	2EAFS	173.16M	44.9%	4.35M	70.6%	73.83→73.64	-0.19
	CPMC[53]	162.00M	48.4%	4.80M	67.5%	73.80→73.01	-0.49
	2EAFS	152.65M	51.4%	3.73M	74.8%	73.83→73.15	-0.68
ResNet-56	G-pruning[68]	87.10M	32.9%	N/A	30%	71.48→70.81	-0.67
	2EAFS	80.00M	36.8%	0.47M	45.4%	71.52→70.60	-0.92
	MIL[69]	76.30M	39.3%	N/A	N/A	71.33→68.37	-2.96
	BNP[70]	67.20M	51.1%	N/A	N/A	72.93→70.07	-2.86
	SFP[62]	59.40M	52.6%	N/A	N/A	71.40→68.79	-2.61
	2EAFS	61.00M	51.8%	0.36M	58.5%	71.52→69.65	-1.87
ResNet-110	**2EAFS**	159.98M	37.3%	0.97M	44.1%	73.62→72.85	-0.77
	2EAFS	140.02M	45.1%	0.85M	51.2%	73.62→72.30	-0.78
	2EAFS	119.86M	53.0%	0.73M	58.2%	73.62→71.89	-1.73

Table 6: Pruning results of ResNet-50 on ILSVRC-2012. PR is used to express the reduction rate of FLOPs.

Model	FLOPs	PR	Top-1%	±Acc%	Top-5%	±Acc(%)
SSS-26	2.82B	31.9%	76.15→74.18	-1.97	92.96→91.91	-1.05
CP[32]	2.73B	34.1%	76.15→72.30	-3.85	92.96→90.80	-2.16
SFP[62]	2.39B	41.8%	76.15→74.61	-1.54	92.87→92.06	-0.81
SSS-32[36]	2.33B	43.9%	76.12→71.82	-4.30	92.86→90.79	-2.07
2EAFS	2.30B	44.1%	76.15→74.75	-1.40	92.87→92.34	-0.53
White Box [71]	2.22B	45.6%	76.15→75.32	-0.83	92.96→92.43	-0.53
RRBP[72]	1.86B	54.5%	76.15→73.00	-3.15	92.96→91.00	-1.96
GAL[37]	1.84B	55.6%	76.15→71.80	-4.35	92.96→90.82	-2.14
ABCPruner[41]	1.79B	56.6%	76.01→73.52	-2.49	92.96→91.51	-1.45
PSO[73]	1.71B	58.1%	76.15→71.93	-4.22	92.87→91.26	-1.61
2EAFS	1.60B	61.1%	76.15→73.46	-2.98	92.87→91.51	-1.30
Hrank[22]	1.55B	62.6%	76.15→71.98	-4.17	92.96→91.01	-1.95
CLR-RNF-0.44[74]	1.23B	70.1%	76.01→72.67	-3.34	92.96→91.09	-1.87
Hrank[22]	0.98B	76.0%	76.15→69.10	-7.05	92.96→89.58	-3.38
2EAFS	1.00B	75.7%	76.15→70.53	-6.05	92.87→89.85	-3.28

[37], SFP [62], GAL [37], and RRBP [72]). These comparative experiments were conducted using ResNet-50 applied to ILSVRC-2012. Unlike in previous experiments, we report the Top-1 and Top-5 accuracy and performance degradation for 2EAFS under different acceleration conditions, as shown in Table 6. Compared with other advanced methods, 2EAFS consistently achieved better recognition results. Specifically, it reduced model FLOPs to 2.30B while achieving 74.75% Top-1 accuracy and 92.34% Top-5 accuracy. Compared to Hrank [30], a recent state-of-the-art method, 2EAFS achieved better performance (73.46% vs. 71.98% and 70.10% vs. 69.10%) under similar acceleration conditions (62.4% vs. 62.6% and 75.7% vs. 76.0%). Under larger acceleration conditions, performance (2.69% vs. 3.15% by RRBP [72] and 4.35% by [37] in Top-1 accuracy) comparable to RRBP and GAL was still achieved. Compared with the automatic search method, in the case of reducing more FLOPs, 2EAFS can achieve lower accuracy drop on Top-5. The above experimental results demonstrate that our proposed method offers significant advantages in pruning ResNet-50 applied to large-scale datasets.

Fig. 5 shows the number of filters in the first two convolutional layers for each residual block of Resnet-50 under different FLOPs constraints. Each time there is a down-sampling operation, the first two residual blocks in the same stage retain more filters. We suggest that when down-sampling is performed with a convolution of stride 2, more filters must be used to compensate for a loss of information caused by the reduced resolution of feature maps. In addition, it is evident this adaptive pruning algorithm is relatively balanced, avoiding poor information transmission caused by excessive pruning of some layers.

Grad-CAM (Gradient-weighted Class Activation Mapping) is a visualization technique that can help evaluate the network's performance [75].The Grad-CAM for ResNet-50 with 2EAFS is shown in Fig. 6, in which the red regions correspond to a high score for a class and the blue regions represent that the feature is suppressed. All Grad-CAMs support the white crane category, in which the head and belly of the white crane are hot and other objects (e.g., vegetation) are cool. According to the Grad-CAM results, the ResNet-50 model optimized by 2EAFS can more effectively suppress vegetation than the original ResNet-50 while still retaining high scores for the white crane class.

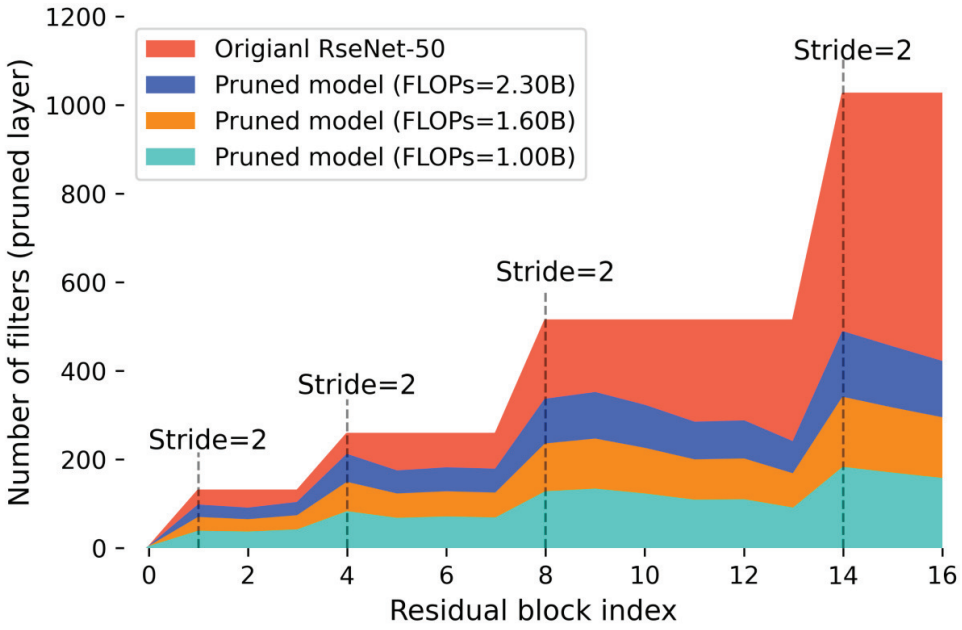
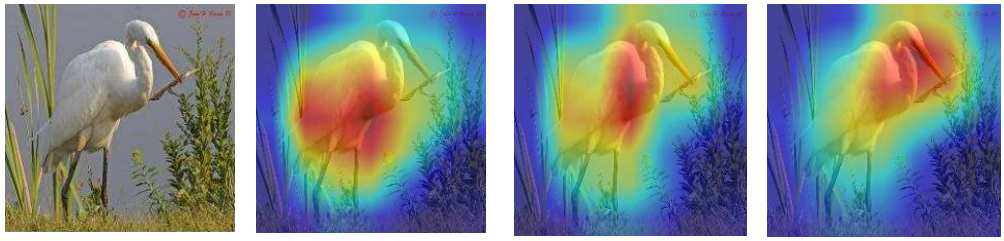


Figure 5: The number of filters in the first two convolutional layers for each residual block of Resnet-50 under different FLOPs constraints.



(a) (b) (c) (d)

Figure 6: Gradient-weighted Class Activation Mapping for Resnet50 with 2EAFS: (a) Original image; (b)Pre-training model; (c) 2EAFS(44.1% reduction in FLOPs); (d) 61.1% reduction in FLOPs.

4.5. Experiments on NWPU-RESISC45 and CUB-200-2011

In order to demonstrate that 2EAFS is also effective in specific scenarios or types of data, we chose typical remote sensing scene and image fine-grained classification tasks for verification, with ResnNet-50 and VGG-16 as benchmark networks, respectively. The experimental results are shown Table 7,

Table 7: The performance of 2EAFS for remote sensing scene recognition and fine-grained image classification

Dataset	Model	Parameters↓	FLOPs↓	Acc(%)	±Acc(%)
NWPU-RESISC45	ResNet-50	0.00%	0.00%	91.53	0.00
		32.19%	30.05%	92.61	+1.08
		51.85%	50.04%	92.06	+0.53
		61.54%	60.02%	91.33	-0.20
CUB-200-2011	VGG-16	0.00%	0.00%	72.32	0.00
		39.92%	32.58%	70.75	-1.57
		43.98%	40.82%	70.06	-2.26

such as: on the remote sensing scene recognition dataset NWPU-RESISC45, with a 50% reduction in FLOPs, a performance improvement of 0.53% is still achieved; for the fine-grained classification dataset CUB-200-2011, with a reduction of about 40% in FLOPs, a loss of 1.57% in accuracy occurs. This is because fine-grained classification tasks require more detailed features than coarse image classification, and compressing the original network model will to some extent lose some of the ability to extract fine-grained features, leading to a certain degree of reduction in the accuracy of fine-grained image classification. However, the accuracy loss is still tolerable.

4.6. Inference speed

To further explore the model performance obtained by 2EAFS pruning, we evaluated the inference speed for four models under different FLOPs. PyTorch implements these tests in CPU(Intel(R) Xeon(R) Silver 4214R CPU @ 2.40GHz) and GPU(GeForce RTX 2080 Ti). In Table 8, we can observe that the pruning procedure can significantly increase the inference speed of the model.

4.7. Discussion

The above comparative experiments demonstrate that 2EAFS can maintain better recognition accuracy while reducing the number of required calculations. We suggest its performance efficacy is primarily derived from adaptive filter search and the inheritance of weight importance.

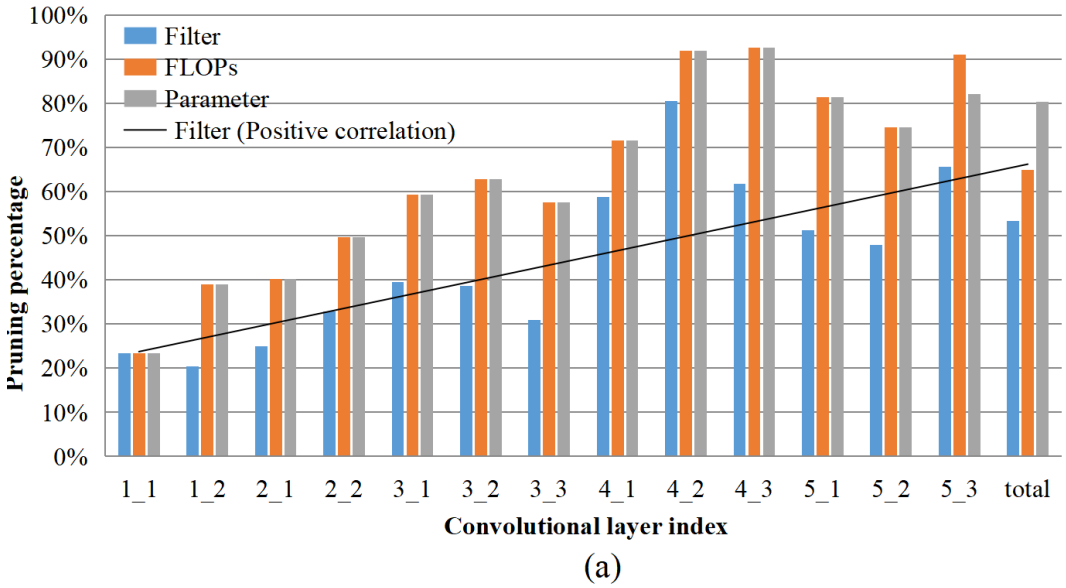
Adaptive Filter Search. Conventional neural network pruning methods based on pretraining-dependency [14, 15, 16] and regularization-retraining [40, 39, 41, 42] require a fixed pruning structure or a series of hyper-parameter

Table 8: FPS of models with different pruning rates on GPU and CPU.

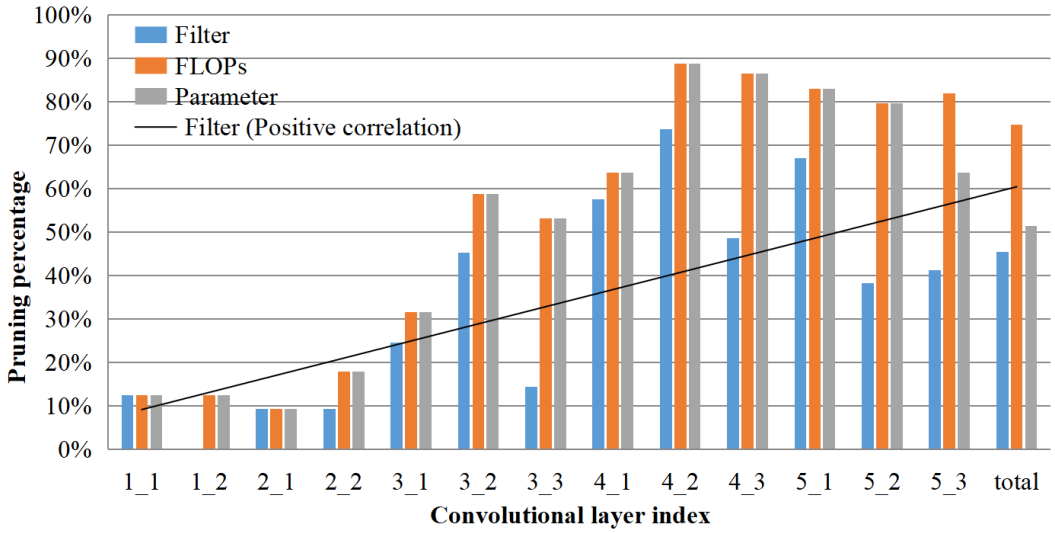
Image size	Model	FLOPs↓	FPS(CPU)	FPS(GPU)
32×32	VGG-16	0.00%	395.39	203.08
		50.0%	411.00	289.34
		74.9%	477.67	364.52
32×32	ResNet-56	0.00%	110.83	119.13
		30.4%	125.77	125.57
		74.7%	131.13	135.42
32×32	ResNet-110	0.00%	56.87	59.18
		43.1%	65.89	66.52
		70.6%	67.85	69.67
224×224	ResNet-50	0.00%	130.05	23.73
		43.1%	135.04	28.46
		58.8%	137.28	33.25

tuning processes, which often produces sub-optimal results. Although automatic subnet searches [40, 39, 41, 42] based on a self-heuristic can provide a better network structure model, higher computational costs are involved that often produce suboptimal search results. The hierarchical pruning structures of VGG-16 with imposed FLOPs constraints are shown in Fig. 7 for different datasets (CIFAR-10 and CIFAR-100). It is evident the pruning rate varies between layers and the parameter pruning ratio is positively correlated with the depth of the convolutional layer. In other words, as the depth increases, more filters tend to be culled. 2EAFS self-adapts to derive a deterministic optimal pruned architecture without human involvement by introducing a measure of convolutional layer importance.

Importance Weight Inheritance. Recent work [65] has demonstrated the essence of filter pruning lies in identifying an optimal pruned architecture rather than selecting the most important filter weights, as done in previous studies [30, 14]. In this subsection, we demonstrate that weights inherited by 2EAFS are sufficiently important using the results of an ablation study. High 2D entropy, low 2D entropy, and random weight inheritance based on pruned networks were extracted using VGG-16 and ResNet-56/110 applied to CIFAR-10. To provide a fair comparison, the other parameters were set to the same values as in previous experiments. It is apparent from the experimental results shown in Table 9 that better performance can be achieved when filters corresponding to high information entropy feature maps undergo weight



(a)



(b)

Figure 7: The pruning rate of each layer for VGG-16 applied to (a) CIFAR-10 and (b) CIFAR-100.

inheritance. Random inheritance also produces a certain degree of performance improvement relative to low information entropy. This indicates the pre-trained network model has "distilled" some critical information through

Table 9: The effects of different weight inheritance methods on the accuracy of pruned networks.

Method	FLOPs	High 2D entropy	Random	Low 2D entropy
VGG-16-P	78.90M	93.17	92.93	92.71
ResNet-56-P	62.45M	93.39	93.11	92.96
ResNet-110-P	74.88M	93.59	93.22	92.83

high 2D entropy, reaffirming the importance of inheriting weight information for recognition accuracy.

5. Conclusion

Existing deep neural networks offer good performance but at high inference and storage costs. As such, this paper proposed an efficient acceleration method of CNNs, 2EAFS, in which information entropy in the feature map (measured by 2D entropy) was used as a theoretical guide to evaluate corresponding filters and compress more compact CNN models. A constraint equation was then constructed based on the negative correlation between filter pruning rates and the importance of convolutional layers. The Nelder-Mead search algorithm was applied to quickly determine the optimal pruning rate in each layer. Finally, the weights of filters were inherited based on the pruning rate and information entropy, and model performance was restored by -tuning. Extensive experiments demonstrated the superiority of 2EAFS compared to other structural pruning algorithms.

6. Declarations

6.1. Conflict of Interest.

All authors disclosed no relevant relationships.

6.2. Data supporting

The datasets generated during and/or analysed during the current study are available from the corresponding author on reasonable request.

6.3. Acknowledgments

We thank LetPub (www.letpub.com) for linguistic assistance and pre-submission expert review.

6.4. Funding Information

This work was supported by NSFC (No.62072489),Henan Science and Technology Innovation Team (CXTD2017091), IRTSTHN (21IRTSTHN013), Zhongyuan Science and Technology Innovation Leading Talent Program (214200510013),Qianjiang laboratory open fund project of Hangzhou Research Institute of Beihang (2020-Y3-A-026).

References

[1] M. Aladem, S. A. Rawashdeh, A single-stream segmentation and depth prediction cnn for autonomous driving, IEEE Intelligent Systems 36 (2020) 79–85.

[2] W. Farag, Z. Saleh, Behavior cloning for autonomous driving using convolutional neural networks (2018) 1–7.

[3] T. Wang, Y. Chen, M. Qiao, H. Snoussi, A fast and robust convolutional neural network-based defect detection model in product quality control, The International Journal of Advanced Manufacturing Technology 94 (2018) 3465–3471.

[4] L. Jiang, Y. Wang, Z. Tang, Y. Miao, S. Chen, Casting defect detection in x-ray images using convolutional neural networks and attention-guided data augmentation, Measurement 170 (2021) 108736.

[5] Z. S. Sabri, Z. Li, Low-cost intelligent surveillance system based on fast cnn, PeerJ Computer Science 7 (2021) e402.

[6] L. Xu, X. Zhou, Y. Tao, L. Liu, X. Yu, N. Kumar, Intelligent security performance prediction for iot-enabled healthcare networks using an improved cnn, IEEE Transactions on Industrial Informatics 18 (2021) 2063–2074.

[7] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 770–778.

[8] X. Zhang, X. Zhou, M. Lin, J. Sun, Shufflenet: An extremely efficient convolutional neural network for mobile devices, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 6848–6856.

[9] K. Han, Y. Wang, Q. Tian, J. Guo, C. Xu, C. Xu, Ghostnet: More features from cheap operations, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 1580–1589.

[10] G. Hinton, O. Vinyals, J. Dean, et al., Distilling the knowledge in a neural network, arXiv preprint arXiv:1503.02531 2 (2015).

[11] C. Li, X. Ma, Z. An, Y. Xu, A deep neural network compression algorithm based on knowledge transfer for edge device, in: 2018 IEEE/ACM Symposium on Edge Computing (SEC), IEEE, 2018, pp. 334–335.

[12] M. Rastegari, V. Ordonez, J. Redmon, A. Farhadi, Xnor-net: Imagenet classification using binary convolutional neural networks, in: European conference on computer vision, Springer, 2016, pp. 525–542.

[13] J. Fang, A. Shafiee, H. Abdel-Aziz, D. Thorsley, G. Georgiadis, J. H. Hassoun, Post-training piecewise linear quantization for deep neural networks, in: European Conference on Computer Vision, Springer, 2020, pp. 69–86.

[14] H. Li, A. Kadav, I. Durdanovic, H. Samet, H. P. Graf, Pruning filters for efficient convnets, arXiv preprint arXiv:1608.08710 (2016).

[15] P. Molchanov, S. Tyree, T. Karras, T. Aila, J. Kautz, Pruning convolutional neural networks for resource efficient inference, arXiv preprint arXiv:1611.06440 (2016).

[16] N. Lee, T. Ajanthan, P. H. Torr, Snip: Single-shot network pruning based on connection sensitivity, arXiv preprint arXiv:1810.02340 (2018).

[17] C. G. Pachón, D. M. Ballesteros, D. Renza, Senpis: Sequential network pruning by class-wise importance score, Applied Soft Computing 129 (2022) 109558. doi:https://doi.org/10.1016/j.asoc.2022.109558.

[18] Z. Wang, X. Xie, G. Shi, Rfpruning: A retraining-free pruning method for accelerating convolutional neural networks, Applied Soft Computing 113 (2021) 107860. doi:https://doi.org/10.1016/j.asoc.2021.107860.

[19] S. Singer, J. Nelder, Nelder-mead algorithm, Scholarpedia 4 (2009) 2928.

[20] Y. LeCun, J. Denker, S. Solla, Optimal brain damage, Advances in neural information processing systems 2 (1989).

[21] S. Srinivas, R. V. Babu, Data-free parameter pruning for deep neural networks, arXiv preprint arXiv:1507.06149 (2015).

[22] Z. Liu, J. Xu, X. Peng, R. Xiong, Frequency-domain dynamic pruning for convolutional neural networks, Advances in neural information processing systems 31 (2018).

[23] X. Dong, S. Chen, S. Pan, Learning to prune deep neural networks via layer-wise optimal brain surgeon, Advances in Neural Information Processing Systems 30 (2017).

[24] C. Chen, F. Tung, N. Vedula, G. Mori, Constraint-aware deep neural network compression, in: Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 400–415.

[25] M. Lin, Y. Zhang, Y. Li, B. Chen, F. Chao, M. Wang, S. Li, Y. Tian, R. Ji, 1xn pattern for pruning convolutional neural networks, IEEE Transactions on Pattern Analysis and Machine Intelligence 45 (2023) 3999–4008. doi:10.1109/TPAMI.2022.3195774.

[26] Y. Ji, L. Liang, L. Deng, Y. Zhang, Y. Zhang, Y. Xie, Tetris: Tile-matching the tremendous irregular sparsity 31 (2018).

[27] A. Zhou, Y. Ma, J. Zhu, J. Liu, Z. Zhang, K. Yuan, W. Sun, H. Li, Learning n: M fine-grained structured sparse neural networks from scratch, arXiv preprint arXiv:2102.04010 (2021).

[28] Y. Zhang, M. Lin, Z. Lin, Y. Luo, K. Li, F. Chao, Y. Wu, R. Ji, Learning best combination for efficient n: M sparsity, arXiv preprint arXiv:2206.06662 (2022).

[29] R. Yu, A. Li, C.-F. Chen, J.-H. Lai, V. I. Morariu, X. Han, M. Gao, C.-Y. Lin, L. S. Davis, Nisp: Pruning networks using neuron importance score propagation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 9194–9203.

[30] M. Lin, R. Ji, Y. Wang, Y. Zhang, B. Zhang, Y. Tian, L. Shao, Hrank: Filter pruning using high-rank feature map, in: Proceedings

of the IEEE/CVF conference on computer vision and pattern recognition, 2020, pp. 1529–1538.

[31] Y. He, P. Liu, Z. Wang, Z. Hu, Y. 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, 2019, pp. 4340–4349.

[32] Y. He, X. Zhang, J. Sun, Channel pruning for accelerating very deep neural networks, in: Proceedings of the IEEE international conference on computer vision, 2017, pp. 1389–1397.

[33] W. Wen, C. Wu, Y. Wang, Y. Chen, H. Li, Learning structured sparsity in deep neural networks, Advances in neural information processing systems 29 (2016).

[34] Z. Liu, J. Li, Z. Shen, G. Huang, S. Yan, C. Zhang, Learning efficient convolutional networks through network slimming, in: Proceedings of the IEEE international conference on computer vision, 2017, pp. 2736–2744.

[35] C. Zhao, B. Ni, J. Zhang, Q. Zhao, W. Zhang, Q. Tian, Variational convolutional neural network pruning, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 2780–2789.

[36] Z. Huang, N. Wang, Data-driven sparse structure selection for deep neural networks, in: Proceedings of the European conference on computer vision (ECCV), 2018, pp. 304–320.

[37] S. Lin, R. Ji, C. Yan, B. Zhang, L. Cao, Q. Ye, F. Huang, D. Doermann, Towards optimal structured cnn pruning via generative adversarial learning, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 2790–2799.

[38] F. Meng, H. Cheng, K. Li, H. Luo, X. Guo, G. Lu, X. Sun, Pruning filter in filter, Advances in Neural Information Processing Systems 33 (2020) 17629–17640.

[39] Z. Liu, H. Mu, X. Zhang, Z. Guo, X. Yang, K.-T. Cheng, J. Sun, Metapruning: Meta learning for automatic neural network channel pruning, in: Proceedings of the IEEE/CVF international conference on computer vision, 2019, pp. 3296–3305.

[40] Y. He, J. Lin, Z. Liu, H. Wang, L.-J. Li, S. Han, Amc: Automl for model compression and acceleration on mobile devices, in: Proceedings of the European conference on computer vision (ECCV), 2018, pp. 784–800.

[41] M. Lin, R. Ji, Y. Zhang, B. Zhang, Y. Wu, Y. Tian, Channel pruning via automatic structure search, arXiv preprint arXiv:2001.08565 (2020).

[42] S. Guo, Y. Wang, Q. Li, J. Yan, Dmcp: Differentiable markov channel pruning for neural networks, in: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2020, pp. 1539–1547.

[43] H. Hu, R. Peng, Y.-W. Tai, C.-K. Tang, Network trimming: A data-driven neuron pruning approach towards efficient deep architectures, arXiv preprint arXiv:1607.03250 (2016).

[44] G. Liu, X. Zheng, Fabric defect detection based on information entropy and frequency domain saliency, The Visual Computer 37 (2021) 515–528.

[45] S. Elkerdawy, M. Elhoushi, A. Singh, H. Zhang, N. Ray, To filter prune, or to layer prune, that is the question (2020).

[46] X. Liu, J. Cao, H. Yao, W. Sun, Y. Zhang, Adapruner: Adaptive channel pruning and effective weights inheritance, arXiv preprint arXiv:2109.06397 (2021).

[47] M. Lin, R. Ji, S. Li, Y. Wang, Y. Wu, F. Huang, Q. Ye, Network pruning using adaptive exemplar filters, IEEE Transactions on Neural Networks and Learning Systems (2021).

[48] A. Krizhevsky, G. Hinton, et al., Learning multiple layers of features from tiny images (2009).

[49] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al., Imagenet large scale visual recognition challenge, International journal of computer vision 115 (2015) 211–252.

[50] G. Cheng, J. Han, X. Lu, Remote sensing image scene classification: Benchmark and state of the art, CoRR abs/1703.00121 (2017). arXiv:1703.00121.

[51] C. Wah, S. Branson, P. Welinder, P. Perona, S. Belongie, The Caltech-UCSD Birds-200-2011 Dataset, Technical Report CNS-TR-2011-001, Computation & Neural Systems, California Institute of Technology, 2011.

[52] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, Going deeper with convolutions, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2015, pp. 1–9.

[53] Y. Yan, R. Guo, C. Li, K. Yang, Y. Xu, Channel pruning via multi-criteria based on weight dependency, in: 2021 International Joint Conference on Neural Networks (IJCNN), IEEE, 2021, pp. 1–8.

[54] Z. Wang, X. Liu, L. Huang, Y. Chen, Y. Zhang, Z. Lin, R. Wang, Model pruning based on quantified similarity of feature maps, arXiv preprint arXiv:2105.06052 (2021).

[55] Y. Li, S. Gu, C. Mayer, L. V. Gool, R. Timofte, 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, pp. 8018–8027.

[56] G. Li, M. Zhang, J. Wang, D. Weng, H. Corporaal, Scwc: Structured channel weight sharing to compress convolutional neural networks, Information Sciences 587 (2022) 82–96.

[57] M. Lin, L. Cao, S. Li, Q. Ye, Y. Tian, J. Liu, Q. Tian, R. Ji, Filter sketch for network pruning, IEEE Transactions on Neural Networks and Learning Systems (2021).

[58] M. K. Lee, S. Lee, S. H. Lee, B. C. Song, Channel pruning via gradient of mutual information for light-weight convolutional neural networks, in: 2020 IEEE International Conference on Image Processing (ICIP), IEEE, 2020, pp. 1751–1755.

[59] F. E. Fernandes Jr., G. G. Yen, Pruning deep convolutional neural networks architectures with evolution strategy, Information Sciences 552 (2021) 29–47. doi:https://doi.org/10.1016/j.ins.2020.11.009.

[60] Z. Zhuang, M. Tan, B. Zhuang, J. Liu, Y. Guo, Q. Wu, J. Huang, J. Zhu, Discrimination-aware channel pruning for deep neural networks (2018) 883–894.

[61] S. Gao, F. Huang, J. Pei, H. Huang, Discrete model compression with resource constraint for deep neural networks (2020).

[62] Y. He, G. Kang, X. Dong, Y. Fu, Y. Yang, Soft filter pruning for accelerating deep convolutional neural networks, arXiv preprint arXiv:1808.06866 (2018).

[63] Y. He, Y. Ding, P. Liu, L. Zhu, H. Zhang, Y. Yang, Learning filter pruning criteria for deep convolutional neural networks acceleration (2020) 2006–2015. doi:10.1109/CVPR42600.2020.00208.

[64] H. Bilal, A. Kumar, B. Yin, Pruning filters with l1-norm and capped l1-norm for cnn compression, Applied Intelligence 51 (2021).

[65] Z. Liu, M. Sun, T. Zhou, G. Huang, T. Darrell, Rethinking the value of network pruning, arXiv preprint arXiv:1810.05270 (2018).

[66] T. Li, B. Wu, Y. Yang, Y. Fan, Y. Zhang, W. Liu, Compressing convolutional neural networks via factorized convolutional filters (2019) 3972–3981. doi:10.1109/CVPR.2019.00410.

[67] N. Aghli, E. Ribeiro, Combining weight pruning and knowledge distillation for cnn compression, in: Computer Vision and Pattern Recognition, 2021.

[68] S. A. Aketi, S. Roy, A. Raghunathan, K. Roy, Gradual channel pruning while training using feature relevance scores for convolutional neural networks, IEEE Access 8 (2020) 171924–171932.

[69] X. Dong, J. Huang, Y. Yang, S. Yan, More is less: A more complicated network with less inference complexity, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 5840–5848.

[70] X. Lu, H. Huang, W. Dong, X. Li, G. Shi, Beyond network pruning: a joint search-and-training approach, in: Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence, 2021, pp. 2583–2590.

[71] Y. Zhang, M. Lin, C.-W. Lin, J. Chen, Y. Wu, Y. Tian, R. Ji, Carrying out cnn channel pruning in a white box, IEEE Transactions on Neural Networks and Learning Systems (2022) 1–10. doi:10.1109/TNNLS.2022.3147269.

[72] Y. Zhou, Y. Zhang, Y. Wang, Q. Tian, Accelerate cnn via recursive bayesian pruning (2019) 3306–3315.

[73] S. Niu, K. Gao, P. Ma, X. Gao, H. Zhao, J. Dong, Y. Chen, D. Shen, Exploiting sparse self-representation and particle swarm optimization for cnn compression, IEEE Transactions on Neural Networks and Learning Systems (2022) 1–13. doi:10.1109/TNNLS.2022.3165530.

[74] M. Lin, L. Cao, Y. Zhang, L. Shao, C.-W. Lin, R. Ji, Pruning networks with cross-layer ranking & k-reciprocal nearest filters, IEEE Transactions on Neural Networks and Learning Systems (2022) 1–10. doi:10.1109/TNNLS.2022.3156047.

[75] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, D. Batra, Grad-cam: Visual explanations from deep networks via gradient-based localization, arXiv preprint arXiv:1610.02391 (2017).

Response to Review Comments

Manuscript ID	ASOC-D-22-04168
Title	An accelerating convolutional neural networks via a 2D entropy based-adaptive filter search method for image recognition

Dear Editors and Reviewers,

We would like to thank you and anonymous reviewers for their time spent in evaluating our manuscript and for their constructive comments. We have carefully revised our manuscript according to the comments, which we sincerely appreciate because they helped us to improve the quality of our manuscript. We have thoroughly considered all the comments of the reviewers and revised our manuscript accordingly, and have tried our best to provide the appropriate answer to every question that the reviewers have raised. In addition, we also carefully proofread the experimental results and the grammar of the paper.

The detailed responses to all the comments are given in the following. The original comments are listed as numbered questions whilst their corresponding responses are also listed.

Look forward to hearing from you.

Yours Sincerely,

Huanyu Li

To Reviewer 1:

Q1.1. What is the theoretical foundation behind the 2D Entropy based-Adaptive Filter Search (2EAFS) method? How does it differ from other methods that have been proposed for CNN pruning?

A1.1:2EAFS is a novel pruning method for convolutional neural networks (CNNs). Compared to traditional pruning methods, 2EAFS employs the concept of 2D image entropy to measure the importance of feature maps corresponding to convolution kernels. Additionally, adaptive filters are utilized to enhance the efficiency and accuracy of pruning. Next, we will provide a detailed explanation of the theoretical foundation of the 2EAFS method and its differences from other CNN pruning methods.

A:Theoretical foundation

(1) ***2D entropy of feature map****. Traditional CNN pruning methods usually use L1 or L2 regularization. However, this method can only explain the importance of filters based on the weight values and cannot measure their contribution to image information. The concept of image entropy addresses the quantification of information and can be used to measure the richness of information in feature maps. But the premise is that the image entropy distribution is robust on the entire data set. Fortunately, through many experiments and observations, it is found that the image entropy distribution meets the above requirements. Calculations based on Shannon's entropy theory can only represent entropy values for a whole image. As such, they cannot represent the spatial characteristics of image pixel distributions, which affects the accuracy of information evaluation[1]. To address this issue, 2D information entropy is introduced to represent spatial distribution characteristics in the feature maps, thus more accurately assessing the contribution of each filter to image information.*

(2) ***Global pruning****. Our inspiration comes from network slimming[2], where the authors use the scaling factor of the batch normalization layer to achieve model compression. However, the original method has a problem in that during global pruning, inconsistent data distributions may result in some layers having zero channels, which leads to the failure of the pruning method. Therefore, in subsequent studies, researchers often normalize the data. Similarly, since the range of entropy values is closely related to the size of feature maps, entropy values in different convolutional layers vary significantly. To this end, we propose a max-min normalization to normalize the 2D entropy of each layer to [0, 1].*

(3) ***Adaptive Filter Search****. The last layer of a CNN tends to provide the best discrimination information and is commonly used for various application tasks. However, intermediate convolutional layers contain far more critical information in practice. Thus, by analyzing the correlation between layers, a more appropriate*

pruning method can be developed to achieve higher recognition accuracy while reducing the required computations [3]. Therefore, 2EAFS automatically determines the pruning rate of each layer from the perspective of the importance of the convolution layer. After obtaining the importance evaluation index of each convolution layer, we construct the evaluation function through the negative correlation between the pruning rate and the importance and determine the final pruning rate through Nelder-Mead.

B:Differences with other CNN pruning methods.

(1) ***Pruning strategy.*** *2EFAS uses an adaptive filter search based on 2D information entropy to determine the importance of each filter. In contrast, traditional CNN pruning methods usually use L1 or L2 regularization or measure importance using filter weight statistics such as mean and standard deviation.*

(2) ***Determination of pruning rate.*** *Several recent studies [4,5] have modeled neural network pruning as an optimal structure search problem, using various intelligent algorithms to determine the pruning rate in each layer. However, the structured search must be repeated when budget requirements and datasets are modified, which requires significant computational resources and runtime. 2EFAS constructs a sparsity constraint equation based on the negative correlation between the filter pruning rate and the importance of the convolutional layer. The Nelder-Mead search algorithm is then adopted to quickly and adaptively determine the optimal pruning architect.*

(3) ***Pruning effect.*** *Compared to traditional CNN pruning methods, 2EAFS can significantly reduce the number of model parameters and computational complexity while maintaining model accuracy. The 2EAFS method achieved better results in experiments than other mainstream pruning methods. Meantime, we also chose typical remote sensing scenarios and image fine-grained type recognition tasks to verify that 2EAFS is still effective in specific scenarios or types of data*

[1] Liu, G., & Zheng, X. (2021). Fabric defect detection based on information entropy and frequency domain saliency. The Visual Computer, 37(3), 515-528. Springer.

[2] Liu, Z., Li, J., Shen, Z., Huang, G., Yan, S., & Zhang, C. (2017). Learning efficient convolutional networks through network slimming. In Proceedings of the IEEE international conference on computer vision (pp. 2736-2744).

[3] Chen, S., & Zhao, Q. (2018). Shallowing deep networks: Layer-wise pruning based on feature representations. IEEE Transactions on Pattern Analysis and Machine Intelligence, 41(12), 3048-3056. IEEE.

[4] Liu, Z., Mu, H., Zhang, X., Guo, Z., Yang, X., Cheng, K. -T., & Sun, J. (2019). Metapruning: Meta learning for automatic neural network channel pruning. In

Proceedings of the IEEE/CVF international conference on computer vision (pp. 3296-3305).

[5] Lin, M., Ji, R., Zhang, Y., Zhang, B., Wu, Y., & Tian, Y. (2020). Channel pruning via automatic structure search. arXiv preprint arXiv:2001.08565.

Q1.2:How exactly is the importance of a filter measured using the amount of information contained in feature maps?

A1.2:Thank you very much for the questions raised by the reviewer. In structured pruning, when a filter is removed, the output feature map of that filter is also removed, which indicates the correlation between the filters and feature maps in neural networks. Previous studies[1] have used feature map sparsity to remove filters. However, this data-driven approach depends heavily on the distribution of input images. It also requires large sample quantities to make reasonable predictions, which involves a complex and time-consuming training process. Image entropy can be used for quantitative information measurements and represent the richness of feature maps. Entropy distribution is also robust across the entire data set, which has been demonstrated through extensive empirical validation. Therefore, this paper uses the expected value of the 2D entropy of each feature map to determine the importance of the corresponding filter.

In addition, we also carried out some ablation experiments to prove the importance of the filter weight obtained by two-dimensional entropy. High 2D entropy, low 2D entropy, and random weight inheritance based on pruned networks were extracted using VGG-16 and ResNet-56/110 applied to CIFAR-10. To provide a fair comparison, the other parameters were set to the same values as in previous experiments. It is apparent from the experimental results shown in Table1 that better performance can be achieved when filters corresponding to high information entropy feature maps undergo weight inheritance. Random inheritance also produces a certain degree of performance improvement relative to low information entropy. This indicates the pre-trained network model has "distilled" some critical information through high 2D entropy, reaffirming the importance of inheriting weight information for recognition accuracy.

Table1: The effects of different weight inheritance methods on the accuracy of pruned networks

Method	FLOPs	High 2D entropy	Random	Low 2D entropy
VGG-16	78.90	93.17	92.93	92.71
Resnet-56-P	62.45	93.39	93.11	92.96
Resnet-110-P	74.88	93.59	93.22	92.83

[1]Hu H, Peng R, Tai Y W, et al. Network trimming: A data-driven neuron pruning approach towards efficient deep architectures[J]. arXiv preprint arXiv:1607.03250, 2016.

Q1.3:Can the proposed 2EAFS method be applied to other types of neural networks beyond CNNs? If so, how would the method need to be adapted?

A1.3:Thank you very much for the question raised by the reviewer. Currently, as one of the most important algorithms in deep learning, CNNs (CNNs) have achieved great success in computer vision, classification, and other fields. However, there are still major problems that need to be solved, such as high complexity, a large number of parameters, and difficult deployment. Therefore, this paper focuses on proposing an efficient, intuitive, and advanced model compression algorithm specifically for CNNs.

Since other types of neural networks, such as recurrent neural networks and fully connected neural networks, have significant differences in structure and parameter composition compared to CNNs, it is difficult to apply the 2EAFS method to other neural networks directly. However, its ideas can provide inspiration and reference for optimizing other neural networks. For example, one can try to design pruning algorithms specific to the particular structure of different neural networks or achieve better model optimization effects by combining pruning with other model compression methods.

Q1.4: Are there any limitations to the 2EAFS method that the authors did not address? Are there any specific scenarios or types of data for which the method might not be effective?

Q1.4:Thank you very much for the valuable suggestions provided by the author. The 2EAFS method can be effectively applied to model compression tasks of any CNN. In the experimental comparison of this paper, we chose the ILSVRC-2012 dataset, which is the most typical, complex, and has the most categories, to demonstrate that the proposed method still has good compression performance on large-scale datasets. In order to demonstrate that 2EAFS is also effective in specific scenarios or types of data, we chose typical remote sensing scene and image fine-grained classification tasks for verification, with ResnNet-50 and VGG-16 as benchmark networks, respectively. The experimental results are shown Table2, such as: on the remote sensing scene recognition dataset NWPU-RESISC45, with a 50% reduction in FLOPs, a performance improvement of 0.53% is still achieved; for the fine-grained classification dataset CUB-200-2011, with a reduction of about 40% in FLOPs, a loss of 1.57% in accuracy occurs. This is because fine-grained classification tasks require more detailed features than coarse image classification, and compressing the original network model will to some extent lose some of the ability to extract fine-grained features, leading to a certain degree of reduction in the accuracy of fine-grained image classification. However, the accuracy loss is still tolerable.

Revision location: Abstract, page 5, lines 5 to 7; Lines 16 to 11 on page; Page 17, lines 1 to 4; Page 25, lines 2 to 5; Page 26, lines 2-9; Table 7.

Table 2: The performance of 2EAFS for remote sensing scene recognition and fine-grained image classification

Dataset	Model	Parameters ↓	FLOPs ↓	Acc(%)	±Acc(%)
NWPU-RESISC45	ResNet-50	0.00%	0.00%	91.53	0.00
		32.19%	30.05%	92.61	+1.08
		51.85%	50.04%	92.06	+0.53
		61.54%	60.02%	91.33	-0.20
CUB-200-2011	VGG-16	0.00%	0.00%	72.32	0.00
		39.92%	32.58%	70.75	-1.57
		43.98%	40.82%	70.06	-2.26

Q1.5: How does the performance of the 2EAFS-pruned models compare to other methods of model compression, such as quantization or low-rank decomposition?

A1.5:Thank you for the valuable feedback. The performance of 2EAFS pruned models depends on specific tasks, training epochs, and deployment complexity, making it difficult to directly compare with model compression methods such as quantization or low-rank decomposition. We will analyze the unique characteristics of these three methods to highlight the advantages of 2EAFS.

The 2EAFS pruning algorithm has advantages such as simple operation, significant performance improvement, and easy deployment, and it is suitable for all types of CNNs (including 1×1 convolutional kernels). In addition, the pruned models can be further optimized through parameter quantization or low-rank approximation to improve numerical accuracy and reduce computational complexity while maintaining the required accuracy and computational constraints. This approach is versatile and can be combined with other model compression algorithms.

Quantization cannot eliminate the redundant filters inherent in the network structure, and reducing the bit-width of parameters can lead to precision loss. In addition, when quantizing specific bit widths, many existing training methods and hardware platforms are no longer applicable, and dedicated system architectures need to be designed, resulting in limited flexibility.

Low-rank decomposition has problems such as the high cost of matrix decomposition operations, layer-by-layer decomposition not conducive to global parameter compression, and requiring a large amount of retraining to achieve convergence. In addition, recent years have seen more and more new networks using 1 × 1 convolutions, which are unsuitable for using low-rank decomposition methods and make it challenging to achieve network compression and acceleration.

Q1.6: What are the specific advantages and disadvantages of the proposed 2EAFS method in comparison to other approaches of network pruning?

A1.6:Thank you very much for your valuable suggestions. We summarize the advantages and disadvantages of the proposed method as follows:

(1) Compared with the unstructured pruning method, 2EFAS has the advantages of good portability, high applicability, simple operation and easy deployment while obtaining better accuracy and compression ratio. Besides that，2EFAS can retain the overall structure of the original model and facilitate integration with existing hardware and software structures.

(2) Compared with the classical pruning methods that use L1 or L2 regularization to measure filter importance, 2EFAS utilizes image information entropy to represent the richness of feature maps, thereby solving the problem of insufficient representation of critical information.

(3) Compared to sparsity-constrained pruning methods, 2EFAS does not necessitate any modification of the loss function or retraining but only entails fine-tuning, resulting in substantial savings in resources and computation time, making it more conducive for industrial application.

(4) Compared to model compression algorithms that utilize intelligent optimization algorithms to determine pruning rates for each layer, 2EFAS can directly obtain the optimal pruned network under sparse constraints. This also avoids the difficulty of restructuring the search when the constraint targets or datasets change, and the process is simple and intuitive.

Similarly, there is significant room for improvement in this method, such as how to combine it with other pruning methods to propose a hybrid pruning model and how to further compress the model by pruning fully connected layers.

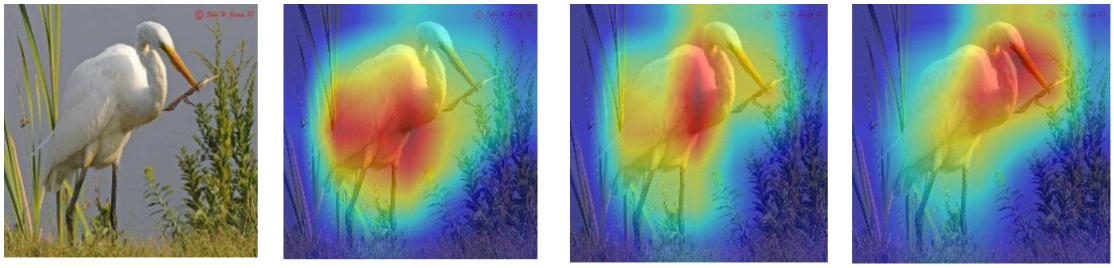
Revision location: Page 6, lines 28 to 30; Page 8, lines 5 to 9; Page 8, lines 26 to 29; Page 9, lines 13 to 16.

Q1.7:What is the impact of the proposed 2EAFS method on the interpretability of the model?

A1.7: Thank you very much for your question. We will use Grad-CAM and ablation experiments to demonstrate the interpretability of 2EAFS.

(1) Grad-CAM (Gradient-weighted Class Activation Mapping) is a visualization technique that can help evaluate the network's performance[1].The Grad-CAM for ResNet-50 with 2EAFS is shown in Fig.1, in which the red regions correspond to a high score for a class and the blue regions represent that the feature is suppressed. All Grad-CAMs support the white crane category, in which the head and belly of the white crane are hot and other objects (e.g., vegetation) are cool. According to the Grad-CAM results, the ResNet-50 model optimized by 2EAFS can more effectively suppress vegetation than the original ResNet-50 while still retaining high scores for

the white crane class.



(a) (b) (c) (d)

Fig. 1: Gradient-weighted Class Activation Mapping for Resnet50 with 2EAFS: (a) Original image; (b)Pre-training model; (c) 2EAFS(44.1\% reduction in FLOPs); (d) 61.1\% reduction in FLOPs.

(2)We interpret that weights inherited by 2EAFS are sufficiently important using the results of an ablation study. High 2D entropy, low 2D entropy, and random weight inheritance based on pruned networks were extracted using VGG-16, and ResNet-56/110 applied to CIFAR-10. The other parameters were set to the same values as in previous experiments to provide a fair comparison. It is apparent from the experimental results shown in Table3 that better performance can be achieved when filters corresponding to high information entropy feature maps undergo weight inheritance. Unexpected inheritance also produces a certain degree of performance improvement relative to low information entropy. This indicates the pre-trained network model has "distilled" some critical information through high 2D entropy, reaffirming the importance of inheriting weight information for recognition accuracy.

Revision location: lines 3 to 12 on page 25; Figure 6

Table 3:The effects of different weight inheritance methods on the accuracy of pruned networks

Method	FLOPs	High 2D entropy	Random	Low 2D entropy
VGG-16	78.90	93.17	92.93	92.71
Resnet-56-P	62.45	93.39	93.11	92.96
Resnet-110-P	74.88	93.59	93.22	92.83

[1] Selvaraju, R. R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., & Batra, D. (2017). Grad-cam: Visual explanations from deep networks via gradient-based localization. In Proceedings of the IEEE international conference on computer vision (pp. 618-626).

Q1.8:Can the proposed 2EAFS method be used in real-time systems?

A1.8: Thank you very much for your questions. Whether it can be applied to real-time systems depends on the model capacity and the specific application scenarios and requirements. Based on normal video playback standard 24FPS, we measured the FPS of the model with the different FLOPs on two chips: CPU (Intel (R) Xeon (R)

Silver 4214CPU @ 2.40 GHz) and GPU (GeForce RTX 2080 Ti). The results are presented in Table 4. As shown, with the increase in pruning rate, the inference speed of the model is significantly improved on both GPU and CPU, surpassing 24FPS and meeting the real-time requirements.

Revision location: Page 26, lines 11 to 16; Table 8.

Table 4. FPS of models with different pruning rates on GPU and CPU.

Image size	Model	FLOPs ↓	FPS(GPU)	FPS(CPU)
32×32	VggNet16	0.00%	395.39	203.08
		50.0%	411.00	289.34
		65.0%	442.56	329.76
		74.9%	477.67	364.52
	ResNet56	0.00%	110.83	119.13
		30.4%	125.77	125.57
		50.7%	127.49	128.57
		74.7%	131.13	135.42
	VggNet110	0.00%	56.87	59.18
		43.1%	65.89	66.52
		58.85	66.46	67.66
		70.6%	67.85	69.67
224×224	ResNet50	0.00%	130.05	23.73
		43.1%	135.04	28.46
		58.8%	137.28	33.25

Q1.9: How does the proposed 2EAFS method compare to other methods that use pruning in combination with other techniques such as distillation, quantization and weight sharing?

Q1.9:Thank you very much for your valuable suggestions. In the experimental part, we added some empirical comparisons of pruning methods combined with distillation [1,2] or weight sharing [3]. It can be seen from the Table 5 that on the CIFAR-10 and CIFAR-100 datasets, 2EAFS can achieve a better recognition effect under the condition of similar compression. Quantization is to reduce parameter storage and memory occupation by reducing numerical bit width from the point of view of hardware deployment. However, pruning, distillation, and parameter sharing are used to eliminate redundant filters in the network structure from the perspective of software.

Considering the difference in principle, this paper does not compare with the quantization method. After that, we will try to quantify the network after 2EAFS pruning and finally realize the deployment of FPGA devices. Thank you again for your valuable suggestions.

Revision location: Table 2, Table 3, and Table 4.

Table 5: Performance comparison of 2EFAS and pruning methods combined with distillation or weight Sharing

Dataset	Model	Method	Parameters ↓	Flops↓	±Acc(%)
Cifar-10	VggNet16	Scwc(s=0.5)[1]	39.5%	41.9%	+0.07
		2EAFS	50.0%	71.4%	+0.37
		Scwc(s=0.2)[1]	69.4%	70.3%	-0.67
		2EAFS	86.1%	74.9%	-0.38
	Resnet56	Scwc(s=0.1)[1]	33.2%	32.6%	+0.10
		2EAFS	34.1%	30.4%	+0.54
		Scwc-0.05[1]	43.0%	41.8%	-0.04
		2EAFS	50.7%	53.7%	+0.13
	Resnet110	Nima et.al[2]	78%	N/A	-1.27
		2EAFS	71%	70.6%	-0.02
Cifar-100	VggNet16	Hu et.al[3]	N/A	40%	-0.61
		2EAFS	74.8%	51.4%	-0.23

[1] Li, G., Zhang, M., Wang, J., Weng, D., & Corporaal, H. (2022). SCWC: Structured channel weight sharing to compress convolutional neural networks. Information Sciences, 587, 82-96. Elsevier.

[2] Aghli, N., & Ribeiro, E. (2021). Combining weight pruning and knowledge distillation for CNN compression. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 3191-3198).

[3] Hu, Y., Sun, S., Li, J., Wang, X., & Gu, Q. (2018). A novel channel pruning method for deep neural network compression. arXiv preprint arXiv:1805.11394.

To Reviewer 2:

Q2.1:The discussion part of related work could be improved. Sec. 2 misses discussion with middle-level pruning granularity like the Nvidia N:M sparsity [1, 2], block sparsity [3,4], which are also an important branch in network pruning.

A2,1:Thank you very much for the valuable suggestions of the reviewer. Indeed, middle-level pruning is an important research direction, and we have added a discussion about this in the related work section. As follows:

Fine-grained Structured Sparse usually group weight elements into small dense regions and prune them at the granularity of groups. Researchers have proposed various grouping methods to achieve better acceleration performance and higher sparsity. Ji et al. [1]proposed a method to rearrange irregular fine-grained sparsity into structured coarse-grained sparsity to bridge the gap between large sparse models and poorer actual acceleration. Lin et al. [2] proposed a novel pattern of 1 ×N for network pruning that achieves significant CPU acceleration while maintaining high-performance accuracy. Supported by the NVIDIA Ampere Core, N: M sparsity leads to attractive storage and computation efficiency and thus has been extensively studied recently. Zhou et al. [3] proposed the sparse-refined straight-through estimator (SR-STE) and sparse architecture divergence (SAD) to train N: M structured sparse networks, achieving significant acceleration. Zhang et al. [4] proved that N: M learning can be naturally formulated as a combinatorial optimization problem of finding the best combination candidates in a finite collection. Even though the aforementioned methods can achieve good speed and recognition accuracy on GPUs with Ampere architecture support, the inference speed on CPUs and ARM is still limited by hardware constraints.

Revision location: lines 2 to 9 on page 6; Page 6, lines 32 to 34; Page 7, lines 1 to 15.

[1] Ji, Y., Liang, L., Deng, L., Zhang, Y., Zhang, Y., & Xie, Y. (2018). TETRIS: TilE-matching the TRemendous Irregular Sparsity. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, & R. Garnett (Eds.), Advances in Neural Information Processing Systems (Vol. 31). Curran Associates, Inc.

[2] Lin, M., Zhang, Y., Li, Y., Chen, B., Chao, F., Wang, M., Li, S., Tian, Y., & Ji, R. (2023). 1xN Pattern for Pruning Convolutional Neural Networks. IEEE Transactions on Pattern Analysis and Machine Intelligence, 45(4), 3999-4008. doi: 10.1109/TPAMI.2022.3195774.

[3] Zhou, A., Ma, Y., Zhu, J., Liu, J., Zhang, Z., Yuan, K., Sun, W., & Li, H. (2021). Learning N: M fine-grained structured sparse neural networks from scratch. arXiv preprint arXiv:2102.04010.

[4] Zhang, Y., Lin, M., Lin, Z., Luo, Y., Li, K., Chao, F., Wu, Y., & Ji, R. (2022). Learning Best Combination for Efficient N: M Sparsity. arXiv preprint arXiv:2206.06662.

Q2.2:The authors stated that their method does not require retraining. However, there is actually a following fine-tuning phase after pruning (Lines 11, Algorit

hm1). This is self-contradicted and I recommend the authors organize a differe nt claim.

A2.2:Thank you very much for raising this issue. We apologize for not providing a clear description, which may have caused some confusion for the reviewer. In the Introduction, page 4, lines 25-26, we intended to convey that 2EAFS does not need to be retrained from scratch and only requires fine-tuning the model after inheriting the weights. We have made the necessary revisions on page 4, lines 25-26, and deleted this statement, and thank the reviewers for reminding us to pay attention to this point.

Q2.3:The compared methods for ImageNet are outdated. I recommend the authors show comparisons with more advanced methods to make their claim more convincing. Please refer to:

A2.3:Thank you very much for raising this issue. Following your suggestion, w e have added more state-of-the-art methods for comparison on ImageNet to ev aluate the effectiveness of 2EFAS.

Revision location: lines 11 to 15 on page 23; 24 pages, lines 13 to 14; Table 6

Q2.4:There are some typos in the current manuscript. Please fix them. I list so me examples here.
1)In sec 3.3, there are some bugs in the format of references ("Several recent studies [? 35, 37, 38] have ⋯.").
2)In page 13, "figure" is missing in "shown in 3(b)".

A2.4:Thank you for pointing out the errors. We have made the necessary revisions in the new version.

Revision location: Page 12, lines 17 to 18; Page 14, lines 33 to 34

Cover letter

Dear Editor,

We want to submit the manuscript entitled "An accelerating convolutional neural networks via a 2D entropy based-adaptive filter search method for image recognition", which we wish to be considered for publication on Applied Soft Computer.

The success of CNNs for various vision tasks has been accompanied by a significant increase in required FLOPs and parameter quantities, which has impeded the deployment of CNNs on devices with limited computing resources and power budgets. Therefore, it is necessary to compress the CNN model. Applied soft computer is committed to publishing papers on CNN model research [1，2]. However, previous studies on structured pruning primarily rely on designed filters and channel importance criteria, without considering the influence of information entropy (contained in feature maps) on model accuracy, which limits pruning effectiveness.

As such, This paper proposes a simple yet effective 2D Entropy based-Adaptive Filter Search (2EAFS) method for fast CNN acceleration. Specifically, the importance of corresponding filters, measured by utilizing the amount of information contained in feature maps, is employed as a theoretical guide to simplify the complex exhaustive search process. Information entropy is then normalized layer by layer and the resulting value is used to calculate a layer-wise importance score in a single step. Additionally, a sparse constraint equation is constructed based on the negative correlation between filter pruning rates and the importance of convolutional layers. The Nelder-Mead search algorithm is then adopted to quickly and adaptively determine the optimal pruning architecture. Finally, importance weights are inherited using the pruning rate and 2D entropy and model performance are restored through fine-tuning. We selected several datasets and carried out extensive experiments to prove the effectiveness of the proposed method. We would be very happy if the submitted manuscript could be reviewed and considered for publication on Applied Soft Computer.

We are looking forward to hearing from you.

Sincerely yours

Chunlei Li, Ph.D., Prof.

School of Electrical and Information Engineering, Zhongyuan University of Technology,

Zhengzhou, 450007, Henan, China

E-mail:lichunlei1979@zut.edu.cn;

[1] C. G. Pach´on, D. M. Ballesteros, D. Renza, Senpis: Sequential network pruning by class-wise importance score, Applied Soft Computing 129 (2022) 109558. doi:https://doi.org/10.1016/j.asoc.2022.109558.

[2] Z. Wang, X. Xie, G. Shi, Rfpruning: A retraining-free pruning method for accelerating convolutional neural networks, Applied Soft Computing 113 (2021) 107860. doi:https://doi.org/10.1016/j.asoc.2021.107860.

Highlights

1. This paper proposes a simple yet effective **2**D **E**ntropy based-**A**daptive **F**ilter **S**earch (2EAFS) method for fast CNN acceleration, which is illustrated in Fig.1.
2. A novel network pruning method (i.e., 2EAFS) is proposed, adopting 2-D entropy feature evaluation and fast adaptive filter search. This end-to-end pruning framework simplifies complexity without retraining or hyperparameter tuning of the model.
3. An efficient filter importance evaluation criterion is proposed. Specifically, only mini-batches of data are passed to the CNN model to accurately estimate the importance of filters based on the 2-D entropy of the feature maps.
4. A negative correlation between filter pruning rate and convolutional layer importance is utilized to construct sparse constraint equations. An optimal pruning architecture for each layer is then quickly determined using the Nelder-Mead search algorithm without human involvement.
5. Extensive experiments were conducted with CNNs (e.g., VGG and ResNet) applied to datasets such as CIFAR-10/100, ILSVRC-2012, NWPU-RESISC45 and CUB-200-2011. Results demonstrated the effectiveness and efficiency of 2EAFS in reducing FLOPs and parameter requirements..

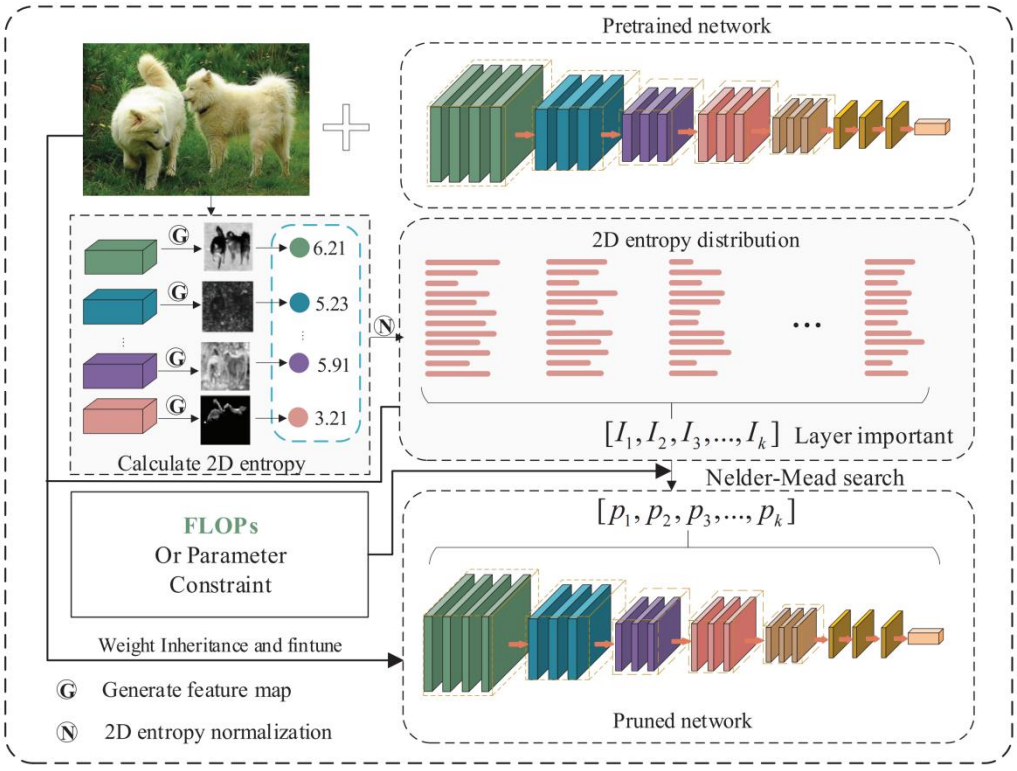


Fig1: The proposed 2EAFS framework. The importance of corresponding filters was first evaluated using the two-dimensional entropy of the feature map. Constraint equations were then constructed using the importance of convolutional layers, while the sub-network structure was adaptively determined by the Nelder-Mead search algorithm. Finally, important weights were inherited based on the pruning rate and 2D entropy and model performance was restored by fine-tuning.

Credit author statement

Chunlei,Li: Conceptualization; Funding acquisition; Resources; Writing—review & editing.

Huanyu,LI: Conceptualization; Methodology; Project administration; Software; Validation; Visualization; Writing—original draft; Writing—review & editing.

Guangshuai Gao: Writing—review & editing.

Zhoufeng Liu: Funding acquisition; Resources.

Pengcheng Liu: Writing—review & editing.

Credit author statement

Chunlei,Li: Conceptualization; Funding acquisition; Resources; Writing—review & editing.
Huanyu,LI: Conceptualization; Methodology; Project administration; Software; Validation; Visualization; Writing—original draft; Writing—review & editing.
Guangshuai Gao: Writing—review & editing.
Zhoufeng Liu: Funding acquisition; Resources.
Pengcheng Liu: Writing—review & editing.

Source Files (word or latex)

Click here to access/download

Source Files (word or latex)

Source Files (latex).zip