FreePrune: An Automatic Pruning Framework Across Various Granularities Based on Training-Free Evaluation

Miao Tang, Ning Liu, *Member, IEEE*, Tao Yang, Haining Fang, Qiu Lin, Yujuan Tan, Xianzhang Chen, *Senior Member, IEEE*, Duo Liu, *Member, IEEE*, Kan Zhong, and Ao Ren

***Abstract*—Network pruning is an effective technique that reduces the computational costs of networks while maintaining accuracy. However, pruning requires expert knowledge and hyperparameter tuning, such as determining the pruning rate for each layer. Automatic pruning methods address this challenge by proposing an effective training-free metric to quickly evaluate the pruned network without fine-tuning. However, most existing automatic pruning methods only investigate a certain pruning granularity, and it remains unclear whether metrics benefit automatic pruning at different granularities. Neural architecture search also studies training-free metrics to accelerate network generation. Nevertheless, whether they apply to pruning needs further investigation. In this study, we first systematically analyze various advanced training-free metrics for various granularities in pruning, and then we investigate the correlation between the training-free metric score and the after-fine-tuned model accuracy. Based on the analysis, we proposed FreePrune score, a more general metric compatible with all pruning granularities. Aiming at generating high-quality pruned networks and unleashing the power of FreePrune score, we further propose FreePrune, an automatic framework that can rapidly generate and evaluate the candidate networks, leading to a final pruned network with both high accuracy and pruning rate. Experiments show that our method achieves high correlation on various pruning granularities and comprehensively improves the accuracy.**

***Index Terms*—Automatic pruning, neural architecture search (NAS), pruning granularities, pruning metric.**

Manuscript received 9 August 2024; accepted 9 August 2024. This work was supported by the Natural Science Foundation of China under Grant 62102051 and Grant 62072059. This article was presented at the International Conference on Compilers, Architectures, and Synthesis for Embedded Systems (CASES) 2024 and appeared as part of the ESWEEK-TCAD Special Issue. This article was recommended by Associate Editor S. Dailey. *(Miao Tang and Ning Liu contributed equally to this work.) (Corresponding author: Ao Ren.)*

Miao Tang, Haining Fang, Qiu Lin, Yujuan Tan, Xianzhang Chen, and Ao Ren are with the College of Computer Science, Chongqing University, Chongqing 400044, China (e-mail: tm@stu.cqu.edu.cn; haining.fang@cqu.edu.cn; linqiu@cqu.edu.cn; tanyujuan@cqu.edu.cn; xzchen@cqu.edu.cn; ren.ao@cqu.edu.cn).

Ning Liu is with Midea Group, Beijing 100102, China (e-mail: ningliu1220@gmail.com).

Tao Yang is with the Department of Computing, The Hong Kong Polytechnic University, Hung Hom, Hong Kong (e-mail: taoyt.yang@connect.polyu.hk).

Duo Liu and Kan Zhong are with the School of Big Data and Software Engineering, Chongqing University, Chongqing 400044, China (e-mail: liuduo@cqu.edu.cn; kzhong@cqu.edu.cn).

Digital Object Identifier 10.1109/TCAD.2024.3443694

I. Introduction

DEEP learning applications have been flourishing in a spectrum of fields, such as computer vision, speech recognition, and natural language processing [1], [2], [3]. However, the explosively increased model size hinders the deployment of deep learning models on embedded devices, which have limited computational and storage resources [4], [5], [6]. To effectively address this problem, model compression emerges as a promising solution [7], [8].

Network pruning is one of the prevailing compression techniques to reduce the computation and storage overhead by reducing the number of model parameters [9]. However, traditional network pruning schemes rely heavily on expert knowledge and involve a cumbersome hyperparameter tuning process, causing significant training costs and deficient pruning rates [10]. To address this challenge, automatic pruning has emerged through modeling the pruning problem as a search process for better-pruned substructures. Fig. 1 illustrates the automatic network pruning pipeline. Based on the original network model, pruning strategies, such as irregular pruning, block pruning, and filter pruning, are applied to specify the sparse patterns of the pruned models. And optimization methods, such as reinforcement learning [11], [12], [13] and evolutionary algorithms [14], [15], [16], [17], [18], are employed to generate a large set of pruned candidates. Next, evaluation metrics are used to evaluate the candidates, aiming at selecting high-quality pruned subnetworks that meet the latency goal and hardware constraints. The final selected network is fine-tuned to obtain the ultimate pruned network.

Among the automatic pruning pipeline, the evaluation metric plays a crucial role in evaluating and selecting the high-quality pruned subnetworks, which can achieve high-pruning rates while maintaining model accuracy. Conventionally, the magnitude is utilized to measure the importance of the weights, where the ones with small magnitudes are deemed redundant and removed [19], [20], [21], [22]. NetworkSlimming [23] adopts the γ in the batch norm (BN) layer as the importance metric, and the filters with small γ values are removed. Besides, entropy-based [24] and KL-divergence-based metrics [25] are also proposed. However, the metrics mentioned above require manually setting the pruning rates for each layer, which causes tremendous training costs and undesired pruning quality. Recent studies have researched

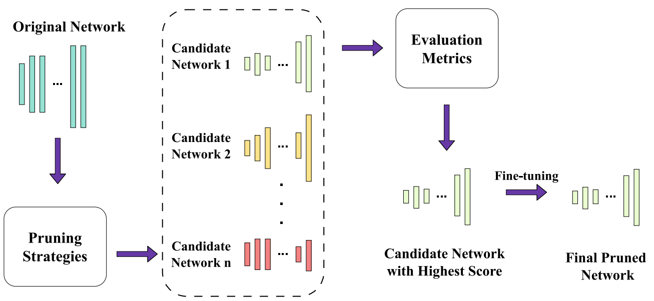

Fig. 1. Pipeline of automatic network pruning.

training-free metrics to skip the laborious hyperparameter tuning and model training process. Eagleeye [26] proposes to update the BN layers with the dataset while not updating the weights, such that the network accuracy can be somewhat recovered and used as an indicator to judge the quality of the pruned subnetworks. Nonetheless, it only evaluates filter pruning, which is prone to failure at high-pruning rates. Synflow [27] has proposed a data-independent iterative synaptic flow pruning, while only unstructured pruning is evaluated. The prior training-free metrics usually only evaluate one pruning granularity, and it prompts us to consider whether there exists a more general metric that can be applied across different pruning granularity scenarios.

Akin to pruning, the neural architecture search (NAS) can also search for a compact network topology [28], [29], [30]. Recent studies on NAS have introduced a range of training-free evaluation metrics. Nonetheless, different from network pruning, the training-free metrics in NAS pay more attention to the characteristics of the overall network structure. For instance, NASWOT [31] constructs training-free metrics based on network expressiveness, and TE-NAS [32] constructs the metric based on network expressiveness and trainability. The connection between network pruning and NAS motivates us to explore how the local characteristics of the weights and the structural property of the network affect pruning, and how they apply to different pruning scenarios.

Drawing from the above, we aim to conduct a systematic evaluation of the applicability of these metrics and construct a more universally applicable training-free metric. Unlike prior studies that focused solely on a single granularity of pruning, we extend our investigation to encompass different granularities to fully unleash the potential of the evaluation metrics. We have investigated the characteristics of the BN layer distribution in pruned networks. Based on the analysis, we further propose FreePrune score, a training-free evaluation metric for rapidly selecting candidate pruning networks according to the distribution of BN statistics. FreePrune score demonstrates a strong correlation with the final accuracy of the pruned subnetworks across various pruning granularities, effectively streamlining the identification of better-pruned structures. Furthermore, to generate a large number of pruned candidates and automate network pruning in conjunction with our proposed metric, we construct an evolutionary algorithm-based automatic pruning framework FreePrune. Meanwhile, a relaxed global pruning technique is employed to initialize the subnetwork population and expedite network evolution.

We summarize our contributions as follows.

1) We systematically analyze and evaluate the mainstream training-free evaluation metrics regarding their applicability on various granularities, and it guides further study on automatic pruning and evaluation metrics.
2) We study the characteristics of the distribution of BN layers in pruned networks and further propose FreePrune score, a training-free metric that can be applied to various pruning granularities and used as a plug-in for rapid evaluation of pruned models.
3) We propose an automatic pruning framework that can effectively compress the search space and generate promising candidate subnetworks, leading to a final high-quality pruned network.
4) Extensive experiments demonstrate that FreePrune score and the pruning framework hold consistent efficiency in searching for high-quality pruned networks.

II. Related Work and Background

A. Granularity for Network Pruning

The effectiveness and applicability of network pruning are influenced by the granularity of the pruning. Additionally, pruning granularity also significantly impacts the deployment of embedded terminals due to the diverse scenarios and requirements. Regarding pruning granularity, pruning techniques can be categorized into unstructured pruning and various structured pruning. Unstructured pruning removes specific neurons in a neural network, while structured pruning follows specific rules to prune the weights of a particular structure. Unstructured pruning lightens neural networks by pruning neurons or connections. For instance, Han et al. [19] utilized weight magnitudes to determine importance, removing connections and weights below a specified threshold, followed by fine-tuning to restore network accuracy. A convex optimization procedure is employed to execute unstructured pruning in [33], aiming to identify sparse subsets within the original weights.

In contrast, structured pruning prunes network weight parameters according to specific rules. For example, He et al. [34] adopted the geometric median for filter pruning, pruning similar redundant filters to minimize similarity between filters. DBP [35] takes a sequence of consecutive layers as a block and removes redundant blocks according to the discrimination of their output features. Unlike previous studies focusing solely on specific types of pruning, our investigation considers diverse pruning scenarios, encompassing both unstructured and various structured pruning to fully explore the potential of pruning and better serve deployment on embedded terminals.

B. Metrics for Pruning and NAS

Evaluation metrics play a pivotal role in the process of selecting pruned subnetwork structures. As the automatic pruning pipeline illustrates, a robust evaluation metric can effectively guide network pruning to select high-quality subnetwork structures. Previous works [21], [23], [34] do not rely on input data and directly evaluate the importance of

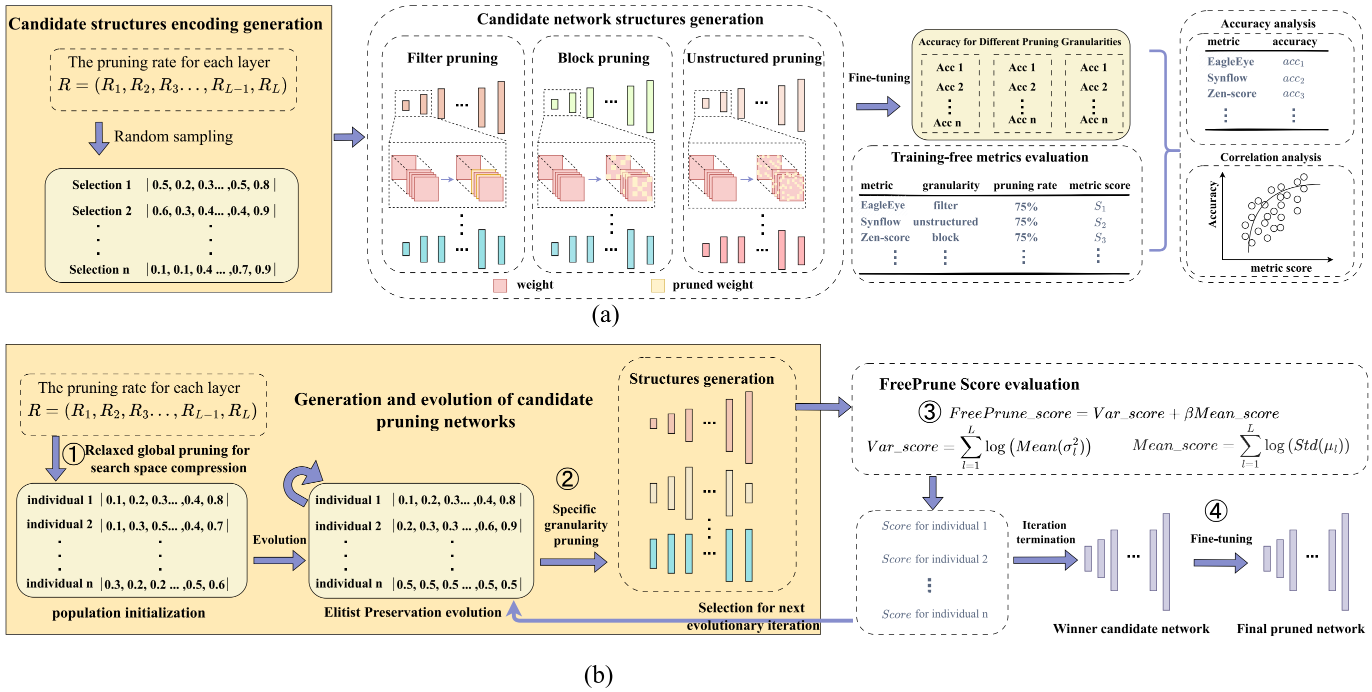

Fig. 2. Overview of the systematic analysis and our proposed framework FreePrune. (a) Systematic analysis. (b) FreePrune framework with FreePrune score.

network structures, usually through regularization methods. On the other hand, works [24], [25], [36], [37] utilize input data for assessing the importance of network structures, typically analyzing gradients, features, and other aspects.

Recent advancements in both automatic pruning and NAS have introduced a range of training-free metrics: while the former focuses on identifying high-quality pruning structures, the latter seeks superior network topologies. However, a notable discrepancy arises in the granularity of metric construction. Automatic pruning typically formulates metrics from the connections between neurons or the neurons themselves, to minimize the perturbation to the original network and thereby attain high-quality sparse substructures. Consequently, it places greater emphasis on the local attributes of the network, such as the fluctuations in neuron gradients and the variations in network accuracy. For example, EagleEye [26] restores the performance of pruned networks by adjusting their batch normalization (BN) layers, leveraging network accuracy as a metric. GraSP [38] and Synflow [27] select the network that can effectively preserve the performance of the original network as the high-quality pruning structure by considering the impact of pruning on network gradients.

On the other hand, NAS endeavors to obtain a high-quality network topology, necessitating a comprehensive consideration of the overarching impact and characteristics of the entire network. For instance, Zen-score [39] utilizes the Gaussian complexity to characterize the number of linear activation regions in the network, thereby representing the expressiveness of the network. NASWOT [31] focuses on the number of representable activation regions in the network and uses Hamming distance to measure the similarity between different inputs, thus constructing a kernel matrix to reflect the network expressiveness. TE-NAS [32] utilizes the number of representable linear activation regions in the network to reflect its expressiveness while utilizing the neural tangent kernel to represent the network trainability.

Given the differences and connections of the training-free metrics in the fields of NAS and pruning, we conducted a systematic analysis and comparative experiments across various pruning granularities to explore their adaptability.

III. Methodology

In this section, we systematically analyze the applicability of various training-free metrics in network pruning across different levels of granularity and pruning rates to explore how the local characteristics of the weights and the structural property of the network affect pruning. Building upon this, we propose a more universally applicable training-free metric, named FreePrune score, and develop a comprehensive automatic pruning framework FreePrune, capable of handling all granularities.

Fig. 2 illustrates systematic analysis and the proposed automatic pruning framework FreePrune. Fig. 2(a) shows the systematic analysis process. Initially, we obtain the pruning configurations of each layer through random sampling to generate the candidate networks with various pruning rates. Subsequently, we execute pruning schemes with various granularities, such as filter pruning, unstructured pruning, and block pruning. The fine-tuning is then performed to produce multiple sets of candidate pruned networks. Finally, we evaluate the effectiveness of the training-free metrics in indicating network performance by measuring the correlation between the metrics and the final network accuracy. Further elaboration on these procedures will be provided in Sections III-A and III-B. Based on our systematic analysis, We discover that the BN layer itself can serve as a reliable indicator for the pruned network performance. We investigate the statistical characteristics of the BN layers of the pruned network and propose a novel and more universally applicable training-free metric FreePrune score. We will show more details in Section III-C.

To further enable automatic pruning under various granularity and pruning rate constraints, we propose an evolutionary algorithm-based pruning framework, which integrates FreePrune score as depicted in Fig. 2(b). Initially, to compress the pruning configuration search space, we propose a relaxed global pruning method to establish the initial pruning configuration. Then, the Elitist Preservation evolutionary algorithm is deployed for the rapid generation and evolution of candidate pruned subnetwork structures. The FreePrune score to efficiently is used to select a better-pruned candidate structure. Finally, at the end of the evolution process, the framework can generate a high-quality pruned subnetwork that satisfies the constraints. More details will be shown in Section III-D.

A. Definition and Network Evolution

Given a CNN model $\mathcal{N}$ with L layers and its parameter set W, where $W = (W_1, W_2, \dots, W_{L-1}, W_L)$, W_l represents the parameters of the l^{th} layer of the model. Let $\mathcal{L}$ represent the loss function of the model and $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{n}$ represent the model dataset, then model pruning can be formulated as

$$W^* = \arg\min_{W^*} \mathcal{L}(\mathcal{N}(W^*), \mathcal{D}), \quad \text{s.t. } \mathcal{C} < \text{Constraints} \tag{1}$$

where $W^* = (W_1^*, W_2^*, \dots, W_{L-1}^*, W_L^*)$ is the subset of the original model parameters. $W_l^* \subset W_l$. $\mathcal{C}$ denotes the constraints satisfied by the model after pruning, such as the number of model parameters and the model inference latency. The objective of pruning is to find an optimal set of substructure parameters that minimize the loss of downstream tasks while satisfying constraints. In this work, we let $R = (R_1, R_2, \dots, R_{L-1}, R_L)$ represent the pruning rates of the model $\mathcal{N}$, where R_l denotes the parameter pruning rate of the model l^{th} layer, and $R_l \subset (0, 1]$, $R_l \subset \mathbb{R}$. This is also the population encoding representation in the evolutionary algorithm, where the pruning rate of each layer of the network is encoded as a real number within a specific range, thus characterizing different pruned subnetworks under specific pruning scenarios. Following the fitness evaluation conducted with our proposed FreePrune score, the better-pruned subnetwork structure is retained for subsequent rounds of evolution.

By conceptualizing network pruning as a search problem, the candidate pruned network structures evolve and generate continuously, guided by FreePrune score, ultimately producing a high-quality pruned network.

B. Systematic Evaluation

To systematically analyze the applicability of training-free metrics in both pruning and the NAS fields, we commence with a series of comparative and analytical experiments across various pruning granularity and pruning rate scenarios. The correlation coefficients and network accuracy are used as the evaluation criteria for the performance of metrics.

For the selection of training-free metrics, we choose typical indicators from both pruning and NAS domains. In the pruning domain, we select the accuracy-based metric EagleEye, as well as the gradient-based Gradnorm and Synflow. In the NAS domain, we choose NASWOT and Zen-score, which reflect the network expressiveness.

The main metrics covered in this article are as follows: 1) *EagleEye* proposes to adjust the BN layer through a small batch of data to restore network performance and uses the adjusted network accuracy as the metric; 2) *Gradnorm* performs network forward propagation through small batches of data and calculates the resulting Euclidean sum of the gradients as the metric; 3) *Synflow* proposes to iteratively preserve the synaptic flow while avoiding layer collapse and employs synaptic saliency score as the measure of network performance; 4) *NASWOT* utilizes the count of activated regions in a neural network to signify network expressiveness and proposes a kernel matrix as the metric by computing the Hamming distance on the activation of the hidden layer; and 5) *Zen-score* measures the expressive capability of a network based on the expectation of Gaussian complexity and employs a scaling factor to address the issue of scale sensitivity.

To assess the effectiveness of the training-free metrics, we use the Spearman and Kendall correlation coefficients. These rank correlation measures indicate monotonic relationships and can efficiently quantify the correlation between metric scores and the final accuracy of the pruned network.

To investigate the impact of pruning granularity and rates, we first analyzed two extreme scenarios: 1) filter pruning and 2) unstructured pruning. We randomly selected 100 sets of candidate pruned networks for CIFAR-10 on VGG under various constraints and calculated the correlation between the evaluation scores of each metric and the final accuracy of the pruned networks after fine-tuning. For the mini-ImageNet with the ResNet18 network, we randomly selected 80 sets of candidate pruned networks.

Table I displays the correlation of each metric under filter pruning and unstructured pruning. For the filter pruning scenario, EagleEye, Zen-score, and NASWOT exhibit high-correlation coefficients, while Synflow demonstrates average correlation coefficients. Conversely, Gradnorm shows a low correlation, indicating its unsuitability as a metric for the filter pruning scenario. For unstructured pruning, EagleEye maintains high-correlation coefficients with Zen-score, while Gradnorm shows poor correlation with NASWOT, suggesting their ineffective application in this scenario.

To further illustrate the indicative effect of each metric, we show the final accuracy of the pruned network resulting from the selection of each metric. As shown in Fig. 3, the upper section displays the accuracy of VGG on the CIFAR-10 dataset, while the lower part displays ResNet18 on the mini-ImageNet dataset. Fig. 3 demonstrates that both EagleEye and Zen-score exhibit a more balanced indication capability when subjected to varying pruning granularities and pruning rates, indicating their potential to identify the high-quality network. Additionally, the average accuracy also reflects the stability of the metric, the higher the correlation coefficient, the better the metric tends to be, and the higher the average accuracy of

TABLE I
CORRELATION COEFFICIENTS OF EACH METRIC AT DIFFERENT PRUNING RATES FOR FILTER PRUNING AND UNSTRUCTURED PRUNING

Granularity	Pruning Rate	Metric	Spearman	Kendall
VGG CIFAR-10 dataset				
filter	75%	EagleEye	0.8172	0.6574
		Zen-score	0.6475	0.4807
		NASWOT	0.8224	0.6512
		Gradnorm	0.0666	0.0381
		Synflow	0.3133	0.2226
	95%	EagleEye	0.8603	0.7060
		Zen-score	0.6834	0.4946
		NASWOT	0.7518	0.5601
		Gradnorm	0.2083	0.1362
		Synflow	0.2125	0.1407
unstructured	75%	EagleEye	0.4884	0.3514
		Zen-score	0.5665	0.4060
		NASWOT	-0.0879	-0.0628
		Gradnorm	-0.1070	-0.0827
		Synflow	0.4273	0.3009
	95%	EagleEye	0.8455	0.6628
		Zen-score	0.8528	0.6653
		NASWOT	0.1458	0.1008
		Gradnorm	-0.4600	-0.3245
		Synflow	0.6869	0.4878
ResNet18 mini-ImageNet dataset				
filter	50%	EagleEye	0.8142	0.6359
		Zen-score	0.7804	0.5755
		NASWOT	0.3186	0.2352
		Gradnorm	0.1212	0.0890
		Synflow	0.4033	0.2639
	75%	EagleEye	0.5939	0.4177
		Zen-score	0.7687	0.5957
		NASWOT	0.4941	0.3415
		Gradnorm	0.1225	0.1032
		Synflow	0.5727	0.4146
unstructured	50%	EagleEye	0.7887	0.6156
		Zen-score	0.5719	0.4132
		NASWOT	0.2524	0.1693
		Gradnorm	-0.2360	-0.1513
		Synflow	0.5883	0.4001
	75%	EagleEye	0.7790	0.6036
		Zen-score	0.5325	0.3844
		NASWOT	0.1772	0.1357
		Gradnorm	-0.3074	-0.2126
		Synflow	0.6156	0.4417

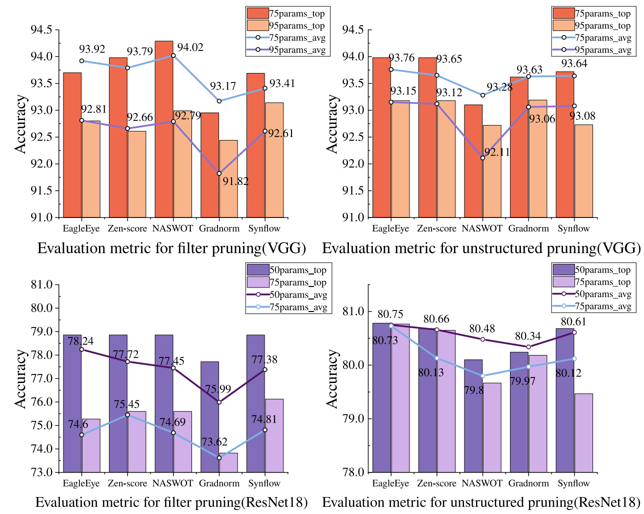

Fig. 3. Final accuracy of the network selected by each metric. The bar represents the accuracy of the best network selected by each metric and the line represents the average accuracy of the top five networks selected by each metric.

TABLE II
CORRELATION COEFFICIENTS OF EACH METRIC AT DIFFERENT PRUNING RATES FOR BLOCK PRUNING IN BLOCK SIZE 16×16 AND 32×32

Granularity	Metric	Pruning Rate	Spearman	Kendall
VGG CIFAR-10 dataset				
block16x16	EagleEye	75%	0.7138	0.5880
		90%	0.6045	0.4370
		95%	0.3167	0.2222
	Zen-score	75%	0.7971	0.6002
		90%	0.8755	0.6905
		95%	0.9338	0.7786
block32x32	Eagleeye	75%	0.8247	0.6286
		90%	0.5959	0.4385
		95%	0.1002	0.0656
	Zen-score	75%	0.8297	0.6572
		90%	0.8972	0.7176
		95%	0.9589	0.8266
ResNet18 mini-ImageNet dataset				
block16x16	EagleEye	50%	0.3478	0.2412
		75%	0.2155	0.1559
	Zen-score	50%	0.6387	0.4681
		75%	0.7158	0.5236
block32x32	Eagleeye	50%	0.2833	0.1821
		75%	0.0106	0.0057
	Zen-score	50%	0.6806	0.5134
		75%	0.7319	0.5291

the resulting candidate networks tends to be. Note that certain instances of *top_acc* values are lower than *avg_acc* because the *top_acc* selected by the indicator is not the highest accuracy.

To explore the applicability of training-free metrics across different pruning granularities, we examine block pruning with varying block sizes. Block pruning, as a relatively fine-grained pruning method within structured pruning, adjusts pruning granularity by varying the size of the pruning blocks. This flexibility enhances its compatibility with hardware platforms in embedded systems, facilitating acceleration and increasing its research value.

We evaluate the EagleEye and Zen-score for various block pruning scenarios, as they have a better correlation in both filter and unstructured pruning, and the accuracy of the resulting network is higher than the other metrics, showing their potential in various pruning granularity scenarios.

Table II shows the correlation of EagleEye and Zen-score for different pruning rates and different block sizes. Zen-score consistently shows high-correlation coefficients, whereas EagleEye exhibits a decrease in correlation coefficients as block size and pruning rate increase, diminishing its predictive effect.

This prompts us to delve into the underlying causes of the differing adaptability between the two metrics, as their main distinction lies in the use of accuracy in EagleEye, a metric reflecting network local characteristics, while Zen-score constructs a macroscopic metric from the perspective

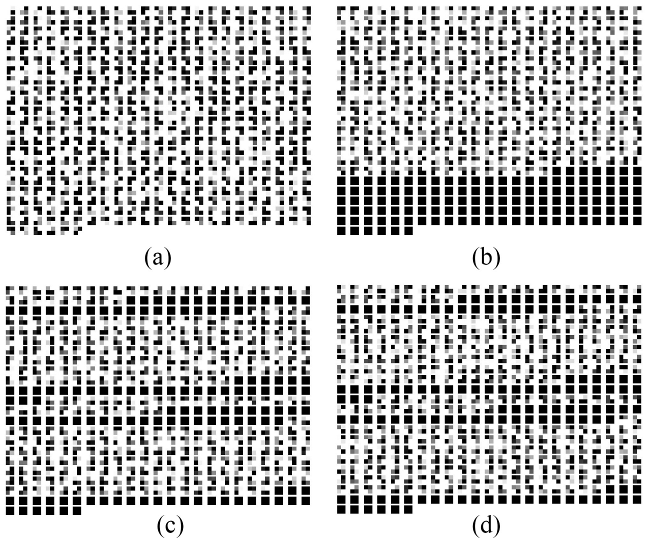

Fig. 4. Last layer feature map of VGG. Where "B64" and "B32" stand for block pruning with size 64×64 and 32×32, respectively. (a) Original feature map. (b) Feature map after B64 pruning and BN adaption. (c) Feature map after block 32×32 pruning. (d) Feature map after B32 pruning and BN adaption.

of network expressiveness. Further experimentation revealed that the primary reason for the failure of EagleEye was the detrimental impact of feature map degradation on network accuracy. As shown in Fig. 4 is the last layer output feature map of VGG on a single sample of the CIFAR-10 dataset. The feature map shows significant damage with a pruning rate of 95% and a block size of 32 × 32 even after BN adaption, and the collapse becomes increasingly apparent as the granularity increases. And this ultimately impacted the network accuracy, showing the limitation of this accuracy-based metric.

Zen-score utilizes the upper bound of Gaussian complexity to measure the number of linear activation classes, which in turn reflects the expressive power of the network. Meanwhile, it employs the variance of the BN layers to mitigate the reduction in discriminative power caused by the BN operations. Using the network expressiveness as a metric gives Zen-score stronger adaptability than EagleEye. However, it fails to thoroughly investigate how the BN layers capture network information.

C. FreePrune Score

EagleEye and Zen-score show better-indication performance than other metrics, it suggests that the BN statistics are promising to serve as the indicators. Inspired by prior studies, we analyze the statistics of the BN layers and propose the FreePrune score. Diverging from the aforementioned approaches, our metric originates from the BN layer itself and takes into account the effects of both mean and variance. We directly assess the ability of the pruned network to capture information through the statistical parameters of the BN layer, thereby formulating the training-free evaluation indicator to effectively reflect the pruning performance.

The statistical parameters of the BN layers include the mean and variance, which are related to the input data of the network, and are computed as (2) for a mini-batch of size N. During the training phase, the above statistical parameters are updated by exponential moving average as in

$$\mu = \frac{1}{N}\sum_{i=1}^{N} x_i,\ \ \sigma^2 = \frac{1}{N-1}\sum_{i=1}^{N}(x_i-\mu)^2 \tag{2}$$

$$\mu_t = m\mu_{t-1} + (1-m)\mu, \sigma_t^2 = m\sigma_{t-1}^2 + (1-m)\sigma^2 \tag{3}$$

where x_i denotes the i_{th} sample in the mini-batch of size N, t denotes the time step, and m signifies the momentum parameter. During the training phase, the variables μ and σ^2 are used to update μ_t and σ_t^2 in the current computation. During the evaluation phase, the BN statistical parameters utilized in the computation are denoted as μ_T and σ_T^2. T denotes the final time step.

As we further investigate the relationship between these statistical parameters and network structure through extensive experiments, we uncover that the distribution of statistical parameters plays a pivotal role in determining the performance of the pruned network and is independent of the input data. Fig. 5 depicts the relationship between the distribution of statistical parameters and the network structure. Fig. 5(a) shows the distribution of the original network, while Fig. 5(b)–(d) show the distribution for block pruning, filter pruning, and unstructured pruning, respectively. Within each type of pruning scenario, the distributions are presented from left to right, showcasing the statistical parameters for the better-pruned network structure followed by the worse-pruned network structure.

It can be observed that after network pruning, the distribution of variance shifts leftward compared to the original network, which represents the loss of network information. However, the better-pruned network structure exhibits a smaller shift, indicating better preservation of the original network information. Regarding the mean values, although they are distributed on both sides of the numerical zero, the better-pruned network structure has a wider range of data distribution compared to the worse-pruned network structure.

To quantify the deviation of the statistical parameters of the BN layer between the pruned and original networks, and to assess the impact of this deviation on network performance, we select 100 groups of pruned networks at each pruning granularity for analysis. Specifically, for variance distance, we first calculate the accumulated difference between the variance of the pruned network and those of the original network across all layers, and then we divide it by the number of channels. For the mean distance, we first calculate the standard deviation of the mean values in both the pruned network and the original network, and then we accumulate the difference between the two standard deviations across all layers. The deviation distance can be formulated as follows:

$$\begin{cases} var\_dis = \sum_{l=1}^{L}\Big(\big(\sigma_l^2\big)^{\text{ori}} - \big(\sigma_l^2\big)^{\text{pruned}}\Big)/c \\ mean\_dis = \sum_{l=1}^{L}\Big(std\big(\mu_l^{\text{ori}}\big) - std\Big(\mu_l^{\text{pruned}}\Big)\Big). \end{cases} \tag{4}$$

The L represents the number of BN layers in the network, $(\sigma_l^2)^{\text{ori}}$ and μ_l^{ori}, $(\sigma_l^2)^{\text{pruned}}$ and μ_l^{pruned} represent the BN statistical parameters of the original network and pruned networks,

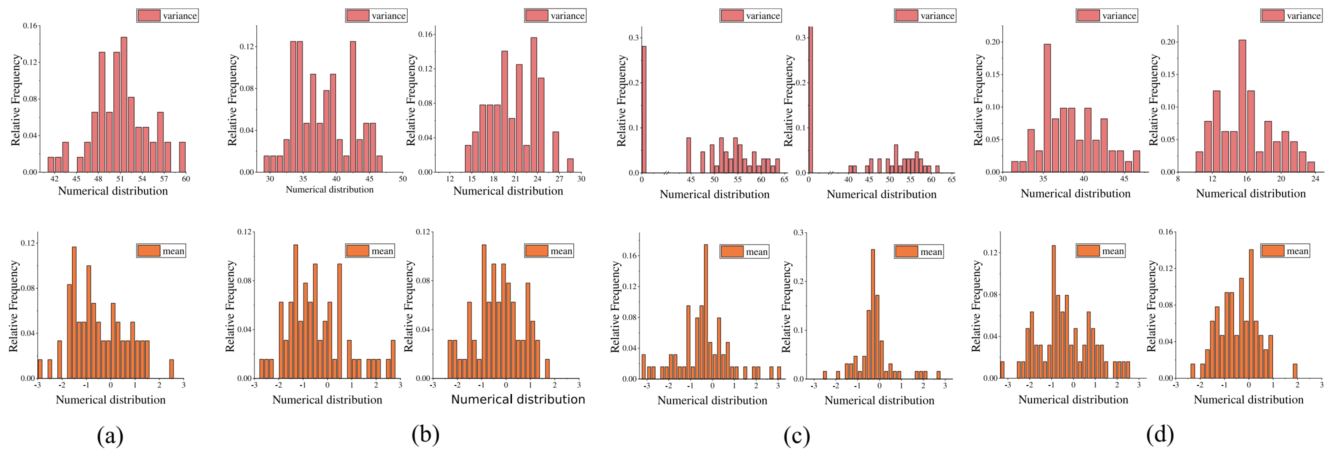

Fig. 5. Histogram of the frequency distribution of mean and variance data for the second BN layer of the VGG network after Gaussian initialization and forward propagation of a batch of data, with the pruning rate of 95% for each granularity. (a) Original network. (b) Network for block pruning. (c) Network for filter pruning. (d) Network for unstructured pruning.

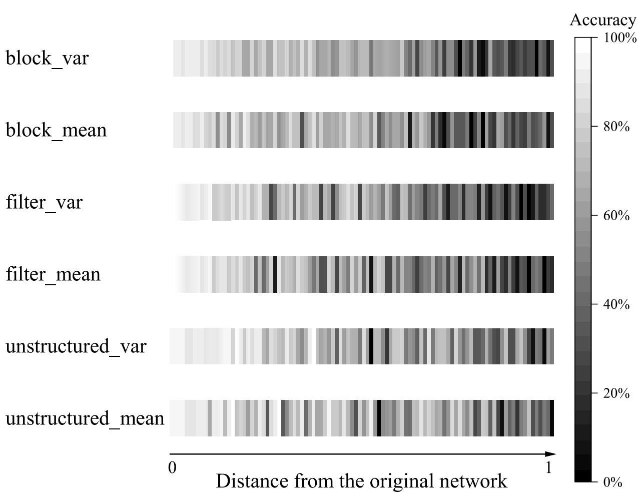

Fig. 6. Visualization of the distances of BN statistics between pruned networks and original network. The horizontal axis represents the normalized distance at different granularities. The color depth indicates the accuracy after normalization at each granularity. The results are obtained using the VGG network with a 95% pruning rate on the CIFAR-10 dataset.

respectively. The c represents the number of channels in the lth layer. The results are presented in Fig. 6, where the color depth represents the different network accuracy. The horizontal axis denotes the cumulative BN deviation distance between the 100 groups of pruned networks and the original network across different pruning granularities, and the distance increases in the direction of the arrow. For ease of comparison, we normalize both the accuracy and the deviation distance at different pruning granularities to show the trend. It can be observed that the smaller the deviation distance between the BN statistical parameters of the pruned network and that of the original network, the better the performance of the pruned network tends to be.

Drawing from the above, the BN layer deviation distance between the pruned network and the original network, which can also be viewed as the degree of loss in information capture ability, demonstrates its potential to indicate the performance of the pruned network. Considering that the BN layer statistical parameters of the original network can be regarded as constants, we simplify the calculation of (4) by only accumulating the BN layer statistical parameters of the pruned network. In this case, the variance distance is equivalent to the cumulative mean of the variances of each BN layer, and the mean distance can be simplified as the cumulative standard deviation of the means of each BN layer. Notably, to enhance discriminability, we employ a logarithmic function to amplify the differences between the layers of the network. Consequently, we propose FreePrune score, which is calculated as

$$FreePrune\_score = Var\_score + \beta Mean\_score$$
$$where \begin{cases} Var\_score = \sum_{l=1}^{L} \log\big(Mean\big(\sigma_l^2\big)\big) \\ Mean\_score = \sum_{l=1}^{L} \log(Std(\mu_l)). \end{cases} \quad (5)$$

The L represents the number of BN layers in the network, σ_l^2 represents the variance of the l^{th} BN, while μ_l represents the mean of the l^{th}. β denotes the balancing parameter, which is set to 0.5 in our experiment. Specifically, we initialize the parameters of the pruned network using Gaussian initialization. Then, we perform forward propagation using a randomly generated batch of data that follows a Gaussian distribution to update the statistical parameters of the BN layers and adjust their distributions. It is worth noting that this process does not involve backpropagation, therefore, it does not include the updating of learnable parameters and does not require the training process of the network, making it achievable at minimal cost. Finally, the FreePrune score is calculated based on (5).

FreePrune score shows a clear correlation between the score and the network trainability. As is illustrated in Fig. 7. Network structures containing greater values of the metrics attain greater accuracy in a reduced number of training iterations, resulting in faster network convergence. This also indicates that our proposed FreePrune score can effectively reflect the trainability of the network under different pruning scenarios.

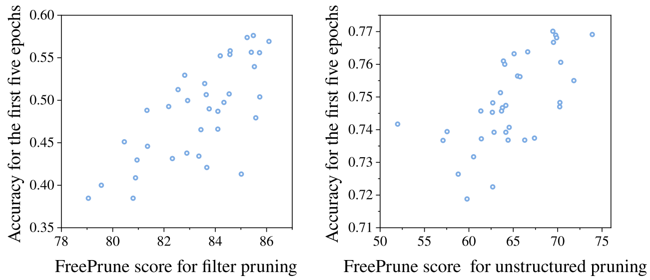

Fig. 7. FreePrune score for network trainability of ResNet18 with a pruning rate of 75%.

Algorithm 1 FreePrune Framework

Input: Target of pruning rate R_p, population size N, iteration round T, original model weights W;

Output: The high-quality pruned subnetwork $\mathcal{N}_*$;

Perform global pruning to determine the pruning rate $R = (R_1, R_2, ..., R_{L-1}, R_L)$ for each layer;
Determine the upper bound R_{ub} and lower bound R_{lb} of the search space according to Eqn. (6);
for each $t \in [1, T]$ **do**
  for each $n \in [1, N]$ **do**
    Perform mutation and crossover with the elitist preservation;
    if Params(n) meets R_p **then**
      Construct candidate subnetwork $\mathcal{N}$ with individual encoding R_n and initialize $\mathcal{N}$ by N(0, 1);
      Perform selection according to Eqn. (5);
    end if
  end for
end for
Obtain the pruned subnetwork encoding with high quality;
Load W to the pruned subnetwork according to Eqn. (1);
Fine-tune the pruned subnetwork until convergence to obtain $\mathcal{N}_*$;
return The high-quality pruned subnetwork $\mathcal{N}_*$

D. FreePrune Framework

To efficiently generate a large number of pruning candidate structures and automate network pruning in conjunction with our proposed metric. We further propose the FreePrune framework. Specifically, FreePrune mainly consists of three components: 1) network structures encoding and search space construction; 2) network structures evolution; and 3) FreePrune score evaluation.

Algorithm 1 illustrates the procedure of our proposed framework. It begins by scaling the search space through relaxed global pruning, as depicted in step 1 of Fig. 2. Subsequently, the Elitist Preservation evolutionary algorithm is employed to evolve candidate pruning structures, which are then filtered using our proposed FreePrune score, as elaborated in steps 2 and 3. Finally, the algorithm returns the high-quality pruned subnetwork that satisfies the constraints, corresponding to step 4 of Fig. 2.

For the structural encoding of the pruned network, we adopt the continuous real number encoding to represent the pruning rates of each layer in the network, which provides a more abundant selection space for pruning candidate structures. Then we determine the feasible upper and lower bounds of the pruning rates to characterize the complete network structure, defining the search space for candidate pruned networks. Specifically, we determine the initial pruning rates $R = (R_1, R_2, \ldots, R_{L-1}, R_L)$ for each layer of the network under a specific total pruning rate using magnitude-based global pruning, where $R_l \subset (0, 1]$. The search space is extremely large, making it difficult to find an optimal starting point and slowing down the search process. This increases the likelihood of encountering local optima. To address these issues, we leverage prior knowledge in network pruning [40] and introduce the relaxed global pruning technique to efficiently reduce the search space. In detail, we introduce a fluctuation ξ above and below this baseline to adjust the upper bound R_{ub} and lower bound R_{lb} of each layer's pruning rate encoding, respectively. The bounds are as follows:

$$\begin{cases} R_{ub} = \min(R + \xi, 1) \\ R_{lb} = \max(R - \xi, R_{\min}). \end{cases} \tag{6}$$

For values exceeding this range, we employ extreme values corresponding to each pruning granularity scenario, such as retaining a minimum of five channels ($R_{\min}$) for filter pruning. This technique effectively saves search overhead. For ξ, we empirically set this value as 30% to balance search efficiency and accuracy in our experiment.

In terms of the network structure evolution, we employ the Elitist Preservation strategy, where the individual with the highest fitness in each generation is preserved as an elite individual while evolving other nonelite individuals. This prevents losing the optimal individual from the current population in the subsequent generation, ensuring global convergence of the genetic algorithm.

The iteration round T is an empirical parameter used to balance search efficiency and final accuracy. If the iteration round concludes without satisfying the pruning rate constraint, the algorithm returns the current best individual. In this case, the individual would re-enter the iteration as a prophetic population individual, allowing for the further search for a constraint-compliant solution at minimal cost based on the previous search. Meanwhile, through the integration of our relaxed global pruning technique with this evolutionary algorithm, the solution can be attained in much fewer iterations. Furthermore, unlike previous approaches that focus on a single granularity of pruning, we integrate pruning schemes for different pruning granularities. This allows for the automatic realization of pruning under multiple pruning granularity scenarios and constraints.

During the fitness evaluation stage, we directly utilize FreePrune score as the criterion for evaluating the fitness of the population. This allows for rapid evaluation and selection of a large number of candidate populations without the need for training. After a specified number of evolutionary rounds, FreePrune score identifies the individual with the highest fitness as the high-quality pruning scheme for the pruned subnetwork. Following specified epochs of network fine-tuning, the final high-quality pruned network can be obtained.

IV. Experiments

In our experiments, we investigate classification downstream task of VGG [41] on CIFAR-10 dataset [42] and ResNet18 [43] on mini-ImageNet dataset [44]. We selected EagleEye [26] and Zen-score [39] for comparison as they performed well in the previous comparative analysis. We visit granularities, including filter pruning, unstructured pruning, and block pruning with sizes 16 × 16 and 32 × 32 with different pruning rates.

A. Experiment Setup

We use the stochastic gradient (SGD) Descent algorithm for fine-tuning with a momentum of 0.9 and the batch size is set to 64. For VGG on CIFAR-10, the weight decay is set to 5e-3 and we fine-tune the network for 150 epochs with a learning rate of 0.0025. For ResNet18 on mini-ImageNet, the learning rate is set to 0.01, and 150 epochs are given for fine-tuning. The same set of candidate pruned networks is used for correlation coefficient evaluation separately. For the automatic pruning framework, we utilize the parameter pruning rates for each pruning granularity as constraint terms. The initial population size is set to 40, and the maximum number of evolution generations is set to 60 for VGG and 80 for ResNet18, respectively. For each pruning scenario, we conducted five experiments to obtain the top pruned network accuracy and average accuracy under each metric. All experiments are implemented on RTX 3090 and Raspberry Pi4.

B. Effectiveness of FreePrune Score

We demonstrate the effectiveness of our proposed metric and framework by utilizing the correlation coefficient and the accuracy of the resulting network from selection as indicators, respectively.

To illustrate the validity of the proposed FreePrune score, we use Spearman and Kendall correlation coefficients to quantify the correlation between our proposed FreePrune score and the final accuracy of the pruned network.

Fig. 8 demonstrates the Spearman and Kendall correlation coefficients for each metric across various pruning scenarios. Fig. 8(a) and (c), respectively, depict the Spearman correlation coefficients for the VGG and ResNet networks under each pruning scenario, while Fig. 8(b) and (d), respectively, showcase the Kendall correlation coefficients for the VGG and ResNet networks under each pruning scenario. The numerical suffixes denote the pruning rate values within each pruning granularity scenario. According to the radar chart, although EagleEye has relatively high-correlation coefficients in some pruning scenarios, they are generally low under the block pruning scenario. This indicates that EagleEye struggles to effectively handle diverse pruning scenarios, particularly those with different structured pruning requirements. This limitation hinders the effective implementation of hardware pruning algorithms. Conversely, our proposed FreePrune score

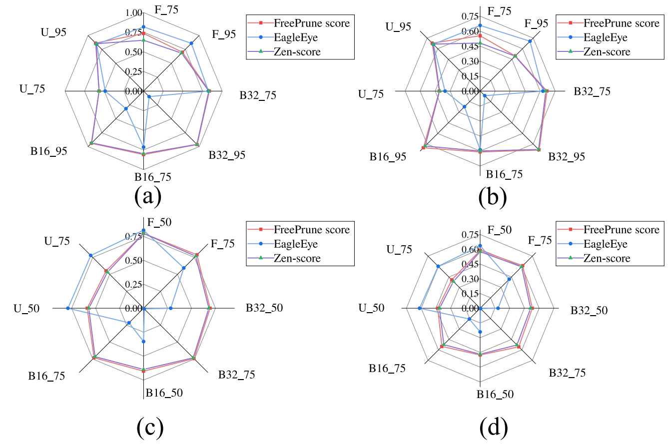

Fig. 8. Radar chart of Spearman and Kendall correlation coefficients for each metric, where “F” stands for filter pruning, “B32” stands for block pruning with size 32×32, “B16” stands for block pruning with size 16×16, and “U” stands for unstructured pruning. (a) Spearman correlation coefficient for VGG. (b) Kendall correlation coefficient for VGG. (c) Spearman correlation coefficient for ResNet18. (d) Kendall correlation coefficient for ResNet18.

demonstrates robust correlation coefficients across various pruning granularities, particularly in the block pruning scenario, demonstrating its adaptability. Moreover, it outperforms Zen-score, with consistently higher values in all pruning scenarios, as shown in the radar chart, where the FreePrune score envelops Zen-score.

To further demonstrate the effectiveness and efficiency of the proposed FreePrune score, we compare the network performance resulting from the selection of pruned networks using each metric. Consistent with the systematic analysis experiments before, for each pruning scenario, we randomly sampled a set of networks and selected the high-quality pruned network using each metric.

Tables III and IV present the comparison of the accuracy of the candidate pruned networks obtained by FreePrune score under different constraints, where *Top_acc* represents the top pruned network accuracy, and *Avg_acc* represents the average accuracy of the top five candidate pruned networks. The proposed FreePrune score consistently tends to select networks with higher accuracy, especially in scenarios with high-pruning rates, demonstrating the effectiveness of our proposed metric. Notably, the average accuracy outperforms other methods, especially in block pruning scenarios. Under other constraints, the accuracy of the pruned network obtained through FreePrune score remains consistent with others.

We also conduct ablation experiments to illustrate the effectiveness of FreePrune. Specifically, we execute the complete automatic pruning framework with a 75% pruning rate on the ResNet18, and then we evaluate and compare the capability of FreePrune and its components in identifying high-quality pruned networks. The results are shown in Table V. The Var_score and Mean_score demonstrate potential in identifying high-quality pruned networks in coarse-grained and fine-grained pruning scenarios, respectively, the proposed FreePrune exhibits superior selecting capabilities across all granularities and consistently outperforms both.

TABLE III
COMPARISON OF THE ACCURACY FOR DIFFERENT METRICS SELECTING NETWORKS WITH VGG ON CIFAR-10

Granularity	Pruning Rate	Metric	Top_acc (%)	Avg_acc (%)
original model	-	-	93.02	-
filter	75%	EagleEye	93.70	**93.92**
		Zen-score	93.70	93.79
		FreePrune score	**93.79**	93.74
	95%	EagleEye	**92.80**	92.81
		Zen-score	92.61	92.66
		FreePrune score	**92.80**	**92.85**
block32x32	75%	EagleEye	**93.62**	93.48
		Zen-score	**93.62**	93.48
		FreePrune score	**93.62**	**93.52**
	95%	EagleEye	83.76	89.26
		Zen-score	92.01	91.91
		FreePrune score	**92.71**	**91.94**
block16x16	75%	EagleEye	**93.93**	93.73
		Zen-score	**93.93**	93.74
		FreePrune score	**93.93**	**93.77**
	95%	EagleEye	91.45	90.70
		Zen-score	**92.39**	**91.33**
		FreePrune score	**92.39**	**91.33**
unstructured	75%	EagleEye	**93.98**	**93.76**
		Zen-score	**93.98**	93.65
		FreePrune score	**93.98**	93.65
	95%	EagleEye	**93.18**	**93.15**
		Zen-score	**93.18**	93.12
		FreePrune score	**93.18**	93.12

TABLE IV
COMPARISON OF THE ACCURACY FOR DIFFERENT METRICS SELECTING NETWORKS WITH RESNET18 ON MINI-IMAGENET

Granularity	Pruning Rate	Metric	Top_acc (%)	Avg_acc (%)
original model	-	-	78.47	-
filter	50%	EagleEye	**78.86**	**78.24**
		Zen-score	**78.86**	77.72
		FreePrune score	**78.86**	77.81
	75%	EagleEye	**75.60**	74.60
		Zen-score	**75.60**	**75.45**
		FreePrune score	**75.60**	**75.45**
block32x32	50%	EagleEye	**76.61**	75.50
		Zen-score	**76.61**	**77.77**
		FreePrune score	**76.61**	**77.77**
	75%	EagleEye	74.36	71.82
		Zen-score	**77.36**	**74.73**
		FreePrune score	**77.36**	**74.73**
block16x16	50%	EagleEye	77.31	76.56
		Zen-score	**77.33**	**77.82**
		FreePrune score	**77.33**	77.76
	75%	EagleEye	77.62	75.53
		Zen-score	**78.49**	76.10
		FreePrune score	**78.49**	**76.79**
unstructured	50%	EagleEye	**80.80**	**80.75**
		Zen-score	80.68	80.66
		FreePrune score	80.78	80.66
	75%	EagleEye	**80.65**	**80.73**
		Zen-score	**80.65**	80.13
		FreePrune score	**80.65**	**80.73**

TABLE V
ABLATION STUDY OF FREEPRUNE ON RESNET18 USING THE MINI-IMAGENET DATASET WITH A 75% PRUNING RATE

Granularity	Metric	Top_acc (%)	Avg_acc (%)
original model	-	78.47	-
filter	Var_score	76.54	76.34
	Mean_score	76.54	75.96
	FreePrune	**76.79**	**76.61**
block32x32	Var_score	77.14	76.49
	Mean_score	77.12	75.62
	FreePrune	**77.48**	**76.83**
block16x16	Var_score	78.15	77.40
	Mean_score	78.20	77.42
	FreePrune	**78.50**	**77.55**
unstructured	Var_score	80.45	80.28
	Mean_score	80.50	80.15
	FreePrune	**80.70**	**80.37**

TABLE VI
COMPARISON OF THE ACCURACY FOR OUR PROPOSED FRAMEWORK SELECTING NETWORKS WITH VGG ON CIFAR-10

Granularity	Pruning Rate	Metric	Top_acc (%)	Avg_acc (%)
original model	-	-	93.02	-
filter	75%	EagleEye	**94.25**	94.02
		Zen-score	94.21	93.97
		FreePrune	94.21	**94.05**
	95%	EagleEye	92.74	92.61
		Zen-score	92.67	92.38
		FreePrune	**92.92**	**92.79**
block32x32	75%	EagleEye	94.07	**93.94**
		Zen-score	93.96	93.82
		FreePrune	**94.25**	**93.94**
	95%	EagleEye	91.53	89.32
		Zen-score	92.48	91.94
		FreePrune	**92.59**	**92.12**
block16x16	75%	EagleEye	**94.14**	93.98
		Zen-score	94.01	93.98
		FreePrune	**94.14**	**94.01**
	95%	EagleEye	92.71	91.66
		Zen-score	92.72	92.51
		FreePrune	**93.06**	**92.79**
unstructured	75%	EagleEye	93.91	**93.86**
		Zen-score	93.95	93.84
		FreePrune	**93.97**	93.84
	95%	EagleEye	93.45	**93.35**
		Zen-score	93.24	93.14
		FreePrune	**93.51**	**93.35**

C. *Effectiveness of FreePrune Framework*

To further demonstrate the effectiveness and efficiency of our proposed FreePrune, we implement the complete framework to find the high-quality pruned subnetwork for different pruning granularity and pruning rate scenarios. To make a fair comparison, we embedded EagleEye and Zen-score into the framework to compare with our proposed method, and for each metric, we conducted five experiments to determine the top accuracy and average accuracy.

As shown in Tables VI and VII, our proposed FreePrune score can effectively obtain pruned networks with higher accuracy in different pruning scenarios, and it has an advantage in the average accuracy of the selected networks, which demonstrates the effectiveness of our proposed method and the automatic pruning framework.

To further demonstrate the superiority of our proposed FreePrune score and the automatic pruning framework, we prune the ResNet18 network on the mini-ImageNet dataset with our framework and randomly sampled pruning configurations, respectively. As illustrated in Fig. 9, our proposed framework improves the accuracy of the pruned network compared to direct random sampling, with a particularly notable improvement observed in scenarios with larger pruning granularity.

TABLE VII
COMPARISON OF THE ACCURACY FOR OUR PROPOSED FRAMEWORK SELECTING NETWORKS WITH RESNET18 ON MINI-IMAGENET

Granularity	**Pruning Rate**	**Metric**	Top_acc **(%)**	Avg_acc **(%)**
original model	-	-	78.47	-
filter	50%	EagleEye	78.76	78.63
		Zen-score	79.13	78.95
		FreePrune	**79.71**	**79.18**
	75%	EagleEye	76.61	76.49
		Zen-score	76.52	76.39
		FreePrune	**76.79**	**76.61**
block32x32	50%	EagleEye	79.85	79.62
		Zen-score	**80.06**	79.78
		FreePrune	79.98	**79.80**
	75%	EagleEye	75.15	72.16
		Zen-score	77.40	76.76
		FreePrune	**77.48**	**76.83**
block16x16	50%	EagleEye	80.05	79.56
		Zen-score	**80.28**	**79.91**
		FreePrune	**80.28**	79.70
	75%	EagleEye	77.72	75.58
		Zen-score	78.23	77.43
		FreePrune	**78.50**	**77.55**
unstructured	50%	EagleEye	81.00	80.88
		Zen-score	81.02	80.84
		FreePrune	**81.15**	**80.94**
	75%	EagleEye	**80.77**	**80.58**
		Zen-score	80.65	80.18
		FreePrune	80.70	80.37

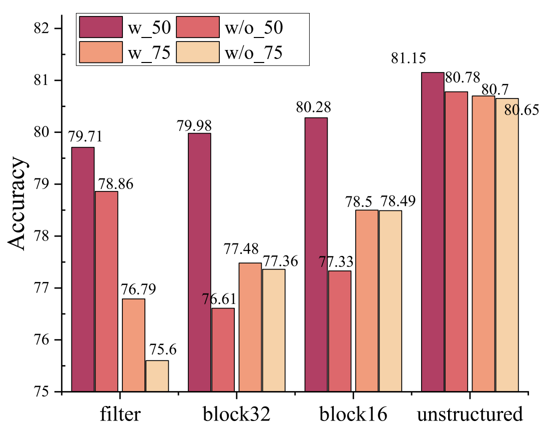

Fig. 9. Comparison of the network accuracy of FreePrune score using the automatic pruning framework and simple random sampling at different pruning granularities. The results are obtained on mini-ImageNet with different pruning rates for ResNet18.

It is noteworthy that by using our proposed method and automatic pruning framework, we can effectively obtain high-quality pruned network structures for various pruning scenarios. Taking filter pruning as an example, for both VGG and ResNet18 networks, accuracy improvements can be achieved while reducing the parameter count by 75% and FLOPs by nearly 60%. Moreover, even at higher-pruning rates, accuracy can be effectively maintained without significant degradation. As shown in Table VIII, we present the performance metrics of the pruned ResNet18 network running on a Raspberry Pi4, with inference time averaged by 100 inference trials. Compared to EagleEye and Zen-score, our proposed automatic pruning framework yields a pruned network with comparable accuracy at a higher-pruning rate, thereby achieving faster inference speeds.

TABLE VIII
PERFORMANCE COMPARISON OF RESNET18 PRUNED NETWORKS ON RASPBERRY PI4 WITH FILTER PRUNING

Metric	**Pruning Rate**	**Acc (%)**	**Inference Time (ms)**
original model	-	78.47	1165.4
EagleEye	75%	75.60	651.5
Zen-score	75%	75.60	651.4
FreePrune	80%	**75.71**	**601.3**

V. CONCLUSION

We systematically evaluated the applicability of mainstream training-free metrics across different pruning granularities and proposed FreePrune score, a training-free metric based on the distribution of BN statistical parameters. Building upon this, we further proposed a comprehensive automatic pruning framework FreePrune, capable of rapidly generating candidate pruned networks and guiding network selection with FreePrune score. FreePrune score demonstrates high correlation across various pruning granularities and pruning rates, making it a reliable tool for rapidly selecting high-quality pruned networks. Extensive experiment results show that FreePrune score and FreePrune framework consistently outperform the prior studies.

ACKNOWLEDGMENT

The authors would like to thank the anonymous reviewers for their valuable comments and improvements to this article.

REFERENCES

[1] S. Dong, P. Wang, and K. Abbas, “A survey on deep learning and its applications,” *Comput. Sci. Rev.*, vol. 40, May 2021, Art. no. 100379.
[2] J. Chai, H. Zeng, A. Li, and E. W. Ngai, “Deep learning in computer vision: A critical review of emerging techniques and application scenarios,” *Mach. Learn. Appl.*, vol. 6, Dec. 2021, Art. no. 100134.
[3] I. Lauriola, A. Lavelli, and F. Aiolli, “An introduction to deep learning in natural language processing: Models, techniques, and tools,” *Neurocomputing*, vol. 470, pp. 443–456, Jan. 2022.
[4] M. M. H. Shuvo, S. K. Islam, J. Cheng, and B. I. Morshed, “Efficient acceleration of deep learning inference on resource-constrained edge devices: A review,” *Proc. IEEE*, vol. 111, no. 1, pp. 42–91, Jan. 2023.
[5] H. Cai et al., “Enable deep learning on mobile devices: Methods, systems, and applications,” *ACM Trans. Design Autom. Electron. Syst.*, vol. 27, no. 3, pp. 1–50, 2022.
[6] M. Dhouibi, A. K. Ben Salem, A. Saidi, and S. Ben Saoud, “Accelerating deep neural networks implementation: A survey,” *Inst. Eng. Technol. Comput. Digit. Techn.*, vol. 15, no. 2, pp. 79–96, 2021.
[7] Z. Li, H. Li, and L. Meng, “Model compression for deep neural networks: A survey,” *Computers*, vol. 12, no. 3, p. 60, 2023.
[8] C.-H. Wang, K.-Y. Huang, Y. Yao, J.-C. Chen, H.-H. Shuai, and W.-H. Cheng, “Lightweight deep learning: An overview,” *IEEE Consum. Electron. Mag.*, vol. 13, no. 4, pp. 51–64, Jul. 2024.
[9] T. Liang, J. Glossner, L. Wang, S. Shi, and X. Zhang, “Pruning and quantization for deep neural network acceleration: A survey,” *Neurocomputing*, vol. 461, pp. 370–403, Oct. 2021.
[10] Y. He, J. Lin, Z. Liu, H. Wang, L.-J. Li, and S. Han, “AMC: AutoML for model compression and acceleration on mobile devices,” in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 784–800.
[11] J. Chen, S. Chen, and S. J. Pan, “Storage efficient and dynamic flexible runtime channel pruning via deep reinforcement learning,” in *Proc. 34th Conf. Neural Inf. Process. Syst.*, 2020, pp. 14747–14758.
[12] Z. Wang and C. Li, “Channel pruning via lookahead search guided reinforcement learning,” in *Proc. IEEE/CVF Winter Conf. Appl. Comput. Vis.*, 2022, pp. 2029–2040.

[13] L. Hirsch and G. Katz, "Multi-objective pruning of dense neural networks using deep reinforcement learning," *Inf. Sci.*, vol. 610, pp. 381–400, Sep. 2022.
[14] L. Lin, S. Chen, Y. Yang, and Z. Guo, "AACP: Model compression by accurate and automatic channel pruning," in *Proc. 26th Int. Conf. Pattern Recognit. (ICPR)*, 2022, pp. 2049–2055.
[15] F. E. Fernandes Jr. and G. G. Yen, "Pruning deep convolutional neural networks architectures with evolution strategy," *Inf. Sci.*, vol. 552, pp. 29–47, Apr. 2021.
[16] C. Pan and X. Yao, "Neural architecture search based on evolutionary algorithms with fitness approximation," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, 2021, pp. 1–8.
[17] Z. Lu et al., "NSGA-Net: Neural architecture search using multi-objective genetic algorithm," in *Proc. Genet. Evol. Comput. Conf.*, 2019, pp. 419–427.
[18] Y. Peng, A. Song, V. Ciesielski, H. M. Fayek, and X. Chang, "PRE-NAS: Evolutionary neural architecture search with predictor," *IEEE Trans. Evol. Comput.*, vol. 27, no. 1, pp. 26–36, Feb. 2023.
[19] S. Han, J. Pool, J. Tran, and W. Dally, "Learning both weights and connections for efficient neural network," in *Proc. 28th Conf. Neural Inf. Process. Syst.*, 2015, pp. 1135–1143.
[20] R. Ao et al., "DARB: A density-adaptive regular-block pruning for deep neural networks," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 5495–5502.
[21] H. Li, A. Kadav, I. Durdanovic, H. Samet, and H. P. Graf, "Pruning filters for efficient ConvNets," in *Proc. Int. Conf. Learn. Represent.*, 2016, pp. 1–13.
[22] Y. He, G. Kang, X. Dong, Y. Fu, and Y. Yang, "Soft filter pruning for accelerating deep convolutional neural networks," in *Proc. 27th Int. Joint Conf. Artif. Intell.*, 2018, pp. 2234–2240.
[23] Z. Liu, J. Li, Z. Shen, G. Huang, S. Yan, and C. Zhang, "Learning efficient convolutional networks through network slimming," in *Proc. IEEE ICCV*, 2017, pp. 2736–2744.
[24] J.-H. Luo and J. Wu, "An entropy-based pruning method for CNN compression," 2017, *arXiv:1706.05791*.
[25] J.-H. Luo and J. Wu, "Neural network pruning with residual-connections and limited-data," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 1458–1467.
[26] B. Li, B. Wu, J. Su, and G. Wang, "Eagleeye: Fast sub-net evaluation for efficient neural network pruning," in *Proc. 16th Eur. Conf. Comput. Vis. (ECCV)*, Glasgow, U.K., 2020, pp. 639–654.
[27] H. Tanaka, D. Kunin, D. L. Yamins, and S. Ganguli, "Pruning neural networks without any data by iteratively conserving synaptic flow," in *Proc. 34th Conf. Neural Inf. Process. Syst.*, 2020, pp. 6377–6389.
[28] B. Zoph and Q. V. Le, "Neural architecture search with reinforcement learning," 2016, *arXiv:1611.01578*.
[29] T. Elsken, J. H. Metzen, and F. Hutter, "Neural architecture search: A survey," *J. Mach. Learn. Res.*, vol. 20, no. 55, pp. 1–21, 2019.
[30] H. Liu, K. Simonyan, and Y. Yang, "Darts: Differentiable architecture search," 2018, *arXiv:1806.09055*.
[31] J. Mellor, J. Turner, A. Storkey, and E. J. Crowley, "Neural architecture search without training," in *Proc. Int. Conf. Mach. Learn.*, 2021, pp. 7588–7598.
[32] W. Chen, X. Gong, and Z. Wang, "Neural architecture search on ImageNet in four GPU hours: A theoretically inspired perspective," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2021, pp. 1–15.
[33] A. Aghasi, A. Abdi, N. Nguyen, and J. Romberg, "Net-trim: Convex pruning of deep neural networks with performance guarantee," in *Proc. 31st Conf. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 1–10.
[34] Y. He, P. Liu, Z. Wang, Z. Hu, and Y. Yang, "Filter pruning via geometric median for deep convolutional neural networks acceleration," in *Proc. IEEE/CVF Conf. CVPR*, 2019, pp. 4340–4349.
[35] W. Wang, S. Zhao, M. Chen, J. Hu, D. Cai, and H. Liu, "DBP: Discrimination based block-level pruning for deep model acceleration," 2019, *arXiv:1912.10178*.
[36] J.-H. Luo, J. Wu, and W. Lin, "ThiNet: A filter level pruning method for deep neural network compression," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 5058–5066.
[37] R. Yu et al., "NISP: Pruning networks using neuron importance score propagation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 9194–9203.
[38] C. Wang, G. Zhang, and R. Grosse, "Picking winning tickets before training by preserving gradient flow," 2020, *arXiv:2002.07376*.
[39] M. Lin et al., "Zen-NAS: A zero-shot NAS for high-performance image recognition," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 347–356.
[40] N. Liu, X. Ma, Z. Xu, Y. Wang, J. Tang, and J. Ye, "Autocompress: An automatic DNN structured pruning framework for ultra-high compression rates," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 4876–4883.
[41] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. 3rd Int. Conf. Learn. Represent. (ICLR)*, 2015, pp. 1–14.
[42] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," Dept. Comput. Sci., Univ. Toronto, Toronto, ON, Canada, Rep. TR-2009, 2009.
[43] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. CVPR*, 2016, pp. 770–778.
[44] O. Vinyals, C. Blundell, T. Lillicrap, K. Kavukcuoglu, and D. Wierstra, "Matching networks for one shot learning," in *Proc. 30th Conf. Neural Inf. Process. Syst.*, 2016, pp. 1–9.