Hybrid Spectral - Gradient Saliency Pruning: A combination of multiple filter scoring criteria

Thanh-Thien Nguyen, Hoang-Loc Tran, Vo-Chi-Dung Nguyen, Viet-An Nguyen, and Duc-Lung Vu

***Abstract*—Deep convolutional neural networks (CNNs) have achieved remarkable performance in visual recognition but remain computationally expensive for deployment on embedded or edge devices. This paper introduces Hybrid Spectral–Gradient Saliency Pruning (HSGSP), a structured pruning framework that unifies spectral analysis and data-driven gradient saliency to achieve efficient CNN compression. The proposed method incorporates a lightweight Frequency Relevance Network (FRN) that learns to estimate the spectral importance of convolutional filters through frequency-band energy ratios, enabling fast, task-driven scoring. A hybrid saliency metric fuses the FRN's spectral relevance with gradient-based Taylor sensitivity, ensuring filters are preserved only when important both spectrally and task-wise. An adaptive iterative schedule dynamically adjusts pruning intensity based on validation feedback, preventing over-pruning and maintaining stability. Experiments on CIFAR-10 and CIFAR-100 using VGG-16BN demonstrate up to 90% parameter reduction with negligible accuracy loss, outperforming recent structured pruning methods. Furthermore, on a Raspberry Pi 5, our pruned model delivers a 3.4× inference speedup while slightly improving accuracy, and when permitting only a 1% accuracy trade-off, the speedup increases dramatically to 7.5×. The results confirm that combining spectral cues with gradient saliency offers a robust and interpretable path toward efficient CNN deployment. The official implementation code of our method is available at https://github.com/locth/HSGSP.git.**

Link to graphical and video abstracts, and to code:
https://latamt.ieeer9.org/index.php/transactions/article/view/10332

***Index Terms*—CNN compression, structured pruning, frequency domain, and model optimization.**

I. Introduction

DEEP convolutional neural networks have transformed computer vision, yet their high computational and memory demands hinder edge deployment. Structured pruning, which removes entire filters or channels, offers practical compression and real speedups without specialized hardware. Existing approaches use various filter-importance criteria [1]–[3], but two issues persist: pruning rates are often fixed and ignore dataset complexity, and most methods focus solely on spatial domains, neglecting frequency-domain information that can reveal filter redundancy [4], [5].

The associate editor coordinating the review of this manuscript and approving it for publication was Giner Alor-Hernández (*Corresponding author: Hoang-Loc Tran*).

T. T. Nguyen, Hoang-Loc Tran, V. C. D. Nguyen, V. A. Nguyen, and D. L. Vu are with University of Information Technology, Vietnam National University Ho Chi Minh City, Vietnam (e-mails: thiennt@uit.edu.vn, locth@uit.edu.vn, 23560012@gm.uit.edu.vn, 23560092@gm.uit.edu.vn, and lungvd@uit.edu.vn).

To address these issues, we propose a new structured pruning framework named Hybrid Spectral–Gradient Saliency Pruning (HSGSP). HSGSP combines a frequency-domain analysis with a data-driven gradient-based saliency in a unified approach. The method adaptively adjusts its pruning schedule based on validation feedback to avoid over-pruning. The key contributions of HSGSP can be summarized as follows:

- Frequency Relevance Network (FRN) for spectral importance: HSGSP introduces a task-driven procedure to learn the spectral importance of filters. A small neural network (FRN) is trained to predict each filter's importance distribution across frequency bands using inputs derived solely from the filter's weights. Weight-domain spectral ratios are computed without data while FRN training uses task signal to adapt dataset characteristic. This learned predictor serves as a proxy to estimate filter relevance without requiring expensive data-driven computations for each filter during pruning.
- Hybrid spectral–gradient saliency score: HSGSP defines a novel importance score that fuses the FRN's frequency-based prediction with the conventional gradient-based saliency (first-order Taylor loss sensitivity). By dynamically weighting the frequency score and gradient score, the method preserves filters that are both spectrally crucial to the loss, thus maintaining both frequency diversity and task relevance in the pruned model.
- Adaptive iterative pruning with validation-aware control: Instead of using a fixed pruning rate or schedule, HSGSP adjusts the fraction of filters pruned in each iteration based on the observed validation loss change. When pruning has little effect on validation loss, the process can continue aggressively; if a significant accuracy drop is detected, the pruning rate is reduced or pruning is halted. This self-correcting mechanism helps prevent overshooting the optimal sparsity level for a given dataset.

Through these innovations, HSGSP aims to achieve higher compression rates without retraining from scratch, and with minimal accuracy loss even on complex datasets. The remainder of this paper is organized as follows. Section 2 reviews related work on structured pruning and frequency-based network analysis. Section 3 details the proposed method, including the FRN design and the hybrid importance score formation, and the pruning mask strategy. Section 4 presents experimental results on CIFAR-10 and CIFAR-100, comparing HSGSP to recent pruning methods on metrics such as accuracy, FLOPs, parameter count, and inference time. Finally,

Section 5 concludes the paper with a summary of contributions and future directions.

II. Related Works

A. Structured CNN Pruning

Structured pruning compresses and accelerates CNNs by removing entire filters or channels. Early methods pruned filters with small weight magnitudes, while Soft Filter Pruning (SFP) [6] enabled filters to regrow during training, and FPGM [1] targeted redundant filters via geometric criteria. Feature-map-based methods like HRank [2] rank filters by output characteristics, and adversarial approaches such as GAL [7] leverage generative networks to replicate original features.

Recent advances refine pruning criteria and schedules. SCOP [8] identifies redundancy using knockoff features, and ResRep [3] achieves nearly lossless pruning via alternating remembering/forgetting phases with fine-tuning, showing the value of iterative pruning [3], [6].

While many pruning approaches rely on manually selected pruning ratios combined with filter-importance criteria, automated ratio-selection methods (e.g., reinforcement learning in AMC or evolutionary search in MetaPruning) can introduce additional optimization complexity. Our method instead adjusts the pruning ratio adaptively during iterative pruning using validation feedback, aiming to balance compression with accuracy recovery across iterations. In contrast to pruning from scratch [9], which trains a sparse/pruned architecture from the beginning, our approach starts from a trained dense model and performs iterative prune and fine-tune cycles, where filters are re-evaluated at each iteration to determine which ones should be retained.

B. Post-training Pruning

Modern deep neural networks often contain substantial redundancy, leading to inefficiencies and deployment challenges on resource-constrained devices. A common compression pipeline is post-training pruning, where pruning is applied to an already trained model and is typically followed by lightweight calibration and/or short fine-tuning to recover performance, depending on the pruning granularity and target compression [10], [11]. In contrast to iterative prune - fine-tune pipelines, post-training approaches usually aim to minimize the number of pruning and retraining cycles by relying on weight-importance metrics or reconstruction-based criteria to sparsify or compress the model efficiently [12].

Recent progress in post-training pruning has introduced several novel frameworks. Kwon et al. [13] developed a fast post-training pruning method for Transformer models that eliminates retraining while significantly lowering computational cost. Frantar et al. [12] integrated pruning and quantization into a unified framework that balances model size and accuracy via precision reduction and parameter elimination. Similarly, Li et al. [14] and Xiao et al. [15] proposed reconstruction-based techniques improving sparsity and neuroregeneration, enhancing the recovery of pruned networks.

Overall, post-training pruning reduces inference overhead and allows efficient deployment on mobile and embedded devices [11], [13]. It eliminates retraining costs, increases flexibility across architectures, and enhances hardware efficiency. However, excessive pruning may degrade accuracy, and certain structural methods depend on specific activation functions or architectures [11]. Moreover, these techniques often require expert tuning and are sensitive to calibration data quality [13].

C. Frequency-Domain Analysis in Pruning

Most pruning methods act in the spatial domain, examining weights or activations. Frequency-domain analysis provides a complementary view based on the spectral interpretation of convolutions. Using Fourier or Cosine transforms, filters or feature maps can be evaluated by their spectral content. Liu et al. [16] introduced Frequency-Domain Dynamic Pruning, removing frequency coefficients during training. Later, Chen et al. [4] applied the Discrete Cosine Transform to retain key filters, and Zhang et al. [5] defined a uniqueness measure preserving distinct spectral patterns. Khaki and Luo [17] merged spatial and frequency saliency in CFDP, improving efficiency without retraining.

These studies show that frequency analysis exposes redundant filters, such as overly smooth or noisy ones. Building on this, our FRN learns filter importance from spectral composition. Instead of fixed heuristics, it uses a small network to predict frequency-wise relevance, trained with teacher signals from filter loss contributions, enabling adaptive identification of essential spectral components across layers and tasks.

D. Sparse Training from Scratch

Sparse-from-scratch training methods are divided into structured and unstructured sparsity. Structured sparsity enforces weight patterns for hardware efficiency, while unstructured sparsity prunes individual weights for flexibility. Unlike post-training pruning, these methods embed sparsity during training, reducing computation and retraining costs [18], [19], though design choices may limit performance.

Recent advances include sparse momentum for efficient gradients [18], orthogonal initialization for stability [19], and dynamic evolutionary sparsity for adaptability [20]. Liu et al. showed sparse RNNs can match dense accuracy [21] and extended sparse training to GANs and random pruning [22], [23].

Sparse training improves efficiency, memory, and inference cost [18], [19], yet poor sparsity selection can harm accuracy, motivating adaptive initialization and weight updates [20]. Structured sparsity suits hardware but needs custom design; unstructured sparsity is flexible but less hardware-aligned. Current work aims to sustain accuracy under extreme sparsity through adaptive and gradient-regenerative techniques.

III. Proposed Method

The proposed HSGSP method consists of two main components: (1) a Frequency Relevance Network (FRN) that learns to estimate filter importance from frequency-domain features, and (2) a hybrid spectral–gradient saliency scoring mechanism that fuses the FRN's output with gradient-based saliency

Algorithm 1 Proposed Frequency-Aware Structured Pruning with FRN Guidance

Input: Pretrained model M, training set $\mathcal{D}_{\text{train}}$, validation set $\mathcal{D}_{\text{val}}$, pruning limit per layer ρ_i, maximum iterations T, accuracy-drop budget $\Delta_{\max}$, initial mixing factor α.

Output: Pruned model $M^{(*)}$ and iteration log $\mathcal{H}$.

Stage 1: Frequency Relevance Network (FRN) Training.

1. Extract spectral features from each convolutional filter using 2D DCT to compute low-, mid-, and high-band energy ratios.
2. Estimate frequency-based saliency targets using first-order Taylor expansion (gradient × weight) across several mini-batches.
3. Train a small MLP (FRN) to map spectral ratios to normalized frequency relevance distributions.
4. Save the trained FRN model f_{FRN} for subsequent scoring.

Stage 2: Iterative Hybrid Pruning.

for $t = 1$ **to** T **do**

(1) Frequency Scoring: For each filter, compute spectral ratios from current weights and obtain frequency relevance scores $f^{(l)}$ via f_{FRN} and adaptive κ_t which is the scalar that selects how much of each filter's DCT spectrum counts as "low frequency" during scoring.

(2) Gradient Scoring: Compute Taylor-based gradient saliency $g^{(l)}$ using a small subset of $\mathcal{D}_{\text{train}}$.

(3) Hybrid Fusion: Normalize both scores and combine them using weighting coefficient α: $s^{(l)} = g^{(l)} \odot \left(f^{(l)} + \varepsilon\right)^{\alpha}$.

(4) Pruning Selection: Remove the lowest-scoring channels per layer under pruning limit ρ_i, keeping at least $C_i^{\min}$ channels.

(5) Model Surgery: Rebuild network by removing pruned channels and adjusting dependent layers.

(6) Fine-Tuning: Retrain the pruned model for a few epochs.

(7) Adaptive Control: Evaluate validation accuracy and update κ_t based on validation loss trend.

if *validation accuracy drop* $> \Delta_{\max}$ **then**

break

end

end

return $M^{(*)}$, $\mathcal{H}$

for pruning decisions. These are integrated into an iterative pruning framework with an adaptive schedule. Algorithm 1 provides a high-level overview of the HSGSP pruning procedure, and the following subsections describe each component in detail.

A. Frequency Relevance Net

The Frequency Relevance Network (FRN) is a learnable model that predicts each convolutional filter's importance based on its frequency features and activations. Acting as a spectral proxy for filter saliency, it estimates a filter's relevance before pruning, helping prioritize those with valuable frequency content and reducing costly calculations.

Unlike traditional pruning that relies on norms or loss impact, FRN learns how a filter's spectral profile relates to task-specific importance. It predicts which frequency bands matter most through a lightweight neural module, shifting pruning from rule-based to data-driven while preserving filters key to generalization.

Formally, FRN maps each filter's spectral feature vector x_j to an importance distribution y_j across B frequency bands (e.g., low, mid, high). The FRN is a function f_{FRN} such that:

$$f_{\text{FRN}} : \mathbf{x}_j \in \mathbb{R}^d \rightarrow \mathbf{y}_j \in [0,1]^B \tag{1}$$

where d is the dimensionality of the input feature vector (typically $d = 3$ in our design). The output y_j is a probability-like distribution over the predefined frequency bands, indicating the relative importance of each band for filter j. We next describe how the training data for the FRN is constructed and how the network is trained.

1) Construction of the FRN Training Dataset: The training dataset for the FRN is automatically derived from a pre-trained CNN model M and a small activation dataset D_a (a subset of images representative of the training data distribution). Each training sample for the FRN corresponds to a single convolutional filter j from M. We obtain both the input features x_j and the target output y_j for each filter through the following steps:

(a) Frequency decomposition: For each convolutional kernel $K \in \mathbb{R}^{H \times W \times C_{\text{in}} \times C_{\text{out}}}$, we first convert it to the frequency domain. A 2D Discrete Cosine Transform (DCT-II) is applied to the spatial dimensions of the kernel, yielding a frequency-domain representation $\hat{K} = \text{DCT2}(K)$. Three frequency bands of interest are defined as: low, mid, and high frequency regions within the $\hat{K}$ spectrum (for example, low-frequency could correspond to the low-frequency coefficients concentrated near the $(0,0)$ position of the DCT).

Denote these band index sets as $\{B_{\text{low}}, B_{\text{mid}}, B_{\text{high}}\} \subset \mathbb{R}^2$, the energy of filter j in band b is computed as the sum of squared magnitudes of the DCT coefficients in that band:

$$E_{b,j} \;=\; \Big\| \{\, \hat{K}_{u,v,j} :\; (u,v) \in B_b \} \Big\|_2^2, \qquad b \in \{\text{low}, \text{mid}, \text{high}\} \tag{2}$$

Here $\hat{K}u, v, j$ denotes the DCT coefficient at frequency position (u, v) for filter j. These energies are normalized into spectral energy ratios. Let $E_{low,j}$, $E_{mid,j}$, $E_{high,j}$ be the energies in each band for filter j. The spectral ratio $r_{b,j}$ for band b is defined as the fraction of the filter's total energy that lies in band b:

$$r_{b,j} \;=\; \frac{E_{b,j}}{\sum_{b' \in \{\text{low, mid, high}\}} E_{b',j}}, \qquad b \in \{\text{low}, \text{mid}, \text{high}\} \tag{3}$$

These three ratios (for low, mid, high) form the FRN's input feature vector x_j.

(b) Taylor-based relevance To create the training target for the FRN, we need a measure of the filter's true importance

in terms of the network's loss. The Taylor-based saliency metric is employed, which estimates how the loss L would change if filter j were removed or altered. Specifically, using a first-order Taylor expansion, the importance T_j of filter j can be approximated by the magnitude of the element-wise product of its gradient and its weight values (also known as the gradient-weight product) averaged over some data samples [2]. Intuitively, T_j measures how much the loss would change if K_j were scaled down slightly (filters with large weights and large gradients contribute more). We further decompose this saliency into the same frequency bands used earlier: using the DCT of the filter, we can attribute portions of T_j to low, mid, and high-frequency components, yielding band-wise contributions $T_{low,j}, T_{mid,j}, T_{high,j}$. To smooth out variations, an exponential moving average (EMA) with decay factor β is applied over these estimates across batches. We then normalize the band-wise importance to form a distribution y_j over the three bands:

$$y_{b,j} \;=\; \frac{\big(T_{b,j}+\varepsilon\big)^{1/\gamma}}{\sum_{b'}\big(T_{b',j}+\varepsilon\big)^{1/\gamma}}, \qquad b,b' \in \{\text{low},\text{mid},\text{high}\} \tag{4}$$

Here $\gamma > 1$ is a sharpening hyper-parameter (in implementaion $\gamma \approx 2.2$ was used) that accentuates the largest components so that y_j is a peaked distribution emphasizing the most critical frequency band for filter j. ε is a small constant to avoid zero. The resulting $y_j = [y_{\text{low},j}, y_{\text{mid},j}, y_{\text{high},j}]$ serves as the target output for the FRN, representing the ideal importance assigned to each frequency band for filter j, according to the Taylor saliency analysis.

2) Architecture and Training of the FRN: The FRN is implemented as a simple multilayer perceptron (MLP) designed to be lightweight yet sufficiently expressive to capture nonlinear relationships between spectral features and importance. In our implementation, the MLP has two hidden layers of moderate size. Specifically, we use:

- Input layer of size $d = 3$.
- Two hidden fully-connected layers with 64 and 32 units respectively, followed by a ReLU activation $\sigma(\cdot)$.
- An output layer with $B = 3$ units (one per frequency band), followed by a softmax function to produce an output distribution $\hat{y} = [\hat{y}_{\text{low}}, \hat{y}_{\text{mid}}, \hat{y}_{\text{high}}]$.

In formula form, the FRN's forward pass can be written as:

$$\hat{y} \;=\; \text{softmax}\Big(W_2\,\sigma(W_1 x + b_1) + b_2\Big) \tag{5}$$

where $W_1 \in \mathbb{R}^{h\times d}$, $W_2 \in \mathbb{R}^{B\times h}$ are weight matrices for the hidden and output layers, b_1, b_2 are bias vectors, and σ is the ReLU activation.

The FRN is trained using the constructed dataset of filters. We use a standard categorical cross-entropy loss between the predicted distribution $\hat{y}_j$ and the target distribution y_j for each filter j. To account for the fact that filters contribute unequally to the network (for instance, filters from earlier layers vs. later layers, or some layers having more filters), one can introduce a weighting factor w_j for each training sample. In our experiments we set $w_j = 1$ for simplicity (equal weighting), but this could be tuned if certain layers needed emphasis. The loss function is:

$$\mathcal{L}_{\text{FRN}} = -\sum_j w_j \sum_b y_{b,j}\log(\hat{y}_{b,j}) \tag{6}$$

After training, the FRN learns to output, for any given filter, a prediction $\hat{y}_j$ that approximates the filter's true frequency-band importance distribution y_j.

3) Inference and Integration into Pruning: Once trained, the FRN becomes a handy tool to estimate frequency importance on the fly. For each filter j (with weights K_j) in the model, we can compute its spectral feature x_j (as in step 1) at any point during pruning, feed it into the FRN, and obtain a predicted distribution $\hat{y}_j$ over frequency bands. We then condense this distribution into a single frequency relevance score S_j for the filter. One simple way is to take a weighted sum of the band importance predictions:

$$S_j = \omega_{\text{low}}\hat{y}_{\text{low},j} + \omega_{\text{mid}}\hat{y}_{\text{mid},j} + \omega_{\text{high}}\hat{y}_{\text{high},j} \tag{7}$$

where $\omega_{\text{low}}, \omega_{\text{mid}}, \omega_{\text{high}}$ are weighting coefficients for the bands. In the simplest case, these ω weights can be all set to 1 (making S_j just the average predicted importance, which still tends to be higher for filters that FRN deems important in any band).

Filters with lower S_j are considered more dispensable (either they lack significant low-frequency content and FRN predicts them as less relevant, or overall their predicted spectral importance is low), whereas filters with higher S_j are deemed important to keep. This FRN-based score S_j will later be combined with the gradient-based score to make final pruning decisions.

B. Hybrid Spectral - Gradient Saliency Score

HSGSP's pruning decision for each filter is based on a hybrid score that fuses the frequency-domain perspective (through the FRN) with the spatial, gradient-based perspective (through the filter's loss gradient). This hybrid score aims to capture both complementary aspects of filter importance: (1) whether the filter carries important frequency information for the task, and (2) whether the filter has a strong influence on the network's output (as measured by gradient saliency). By requiring a filter to be important in both senses to be retained, the method is more selective in pruning truly redundant filters. Fig. 1 demonstrates hybrid scoring which works in two parts:

1) Frequency Score: As described above, for each convolutional layer i and each filter j in that layer, we compute a frequency score $s_{i,j}^{\text{freq}}$ that reflects how much the filter emphasizes low-frequency information and how important that low-frequency content is, according to the FRN. Concretely, let $R_{i,j}^{\text{low}}$ be the low-frequency energy ratio of filter j (from Equation 3), and let $\hat{p}_{i,j}^{\text{low}}$ be the FRN-predicted importance for the low-frequency band of that filter (which is $\hat{y}_{\text{low},j}$ from the FRN output). We define the frequency saliency score for filter j as:

$$s_{i,j}^{\text{freq}} = R_{i,j}^{\text{low}} \times \Big(1 + \hat{p}_{i,j}^{\text{low}}\Big) \tag{8}$$

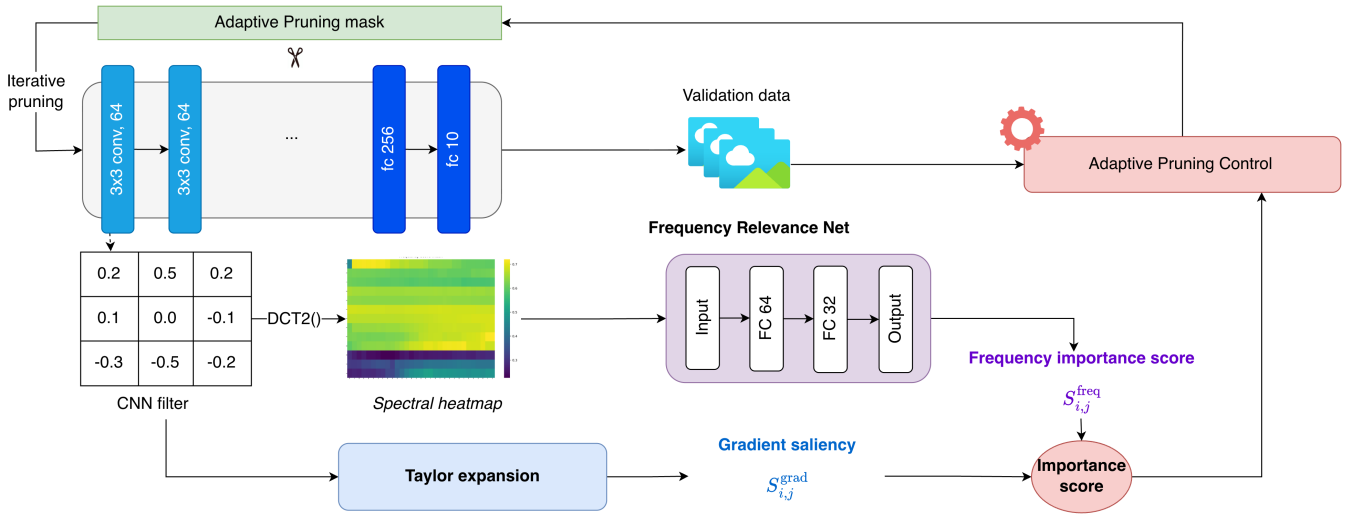

Fig. 1. Overview of the Hybrid Spectral–Gradient Saliency Pruning (HSGSP) framework

Filters with higher low-frequency energy and high FRN-predicted relevance $\hat{p}^{\text{low}}$ receive larger frequency scores $s_{i,j}^{\text{freq}}$. The amplification term $(1+\hat{p}^{\text{low}})$ boosts filters where low-frequency content is important, while those dominated by high-frequency energy get lower scores and are more likely to be pruned. This approach favors filters capturing broad, generalizable patterns over potentially noisy high-frequency features.

2) Gradient Saliency: Alongside the frequency analysis, we compute a gradient-based saliency for each filter, which measures the sensitivity of the network's loss to that filter. We use the classical first-order Taylor criterion, which was given in Eq. (9). In implementation, the network processes a subset of the training data in batches, each mini-batch $m \in \{1, 2, \ldots, M\}$ produces a gradient of the loss $\mathcal{L}^{(m)}$ with respect to each weight $W_{i,k,c,j}^{(l)}$:

$$s_{i,j}^{\text{grad}} = \frac{1}{M} \sum_{m=1}^{M} \sum_{i,k,c} \left| \frac{\partial \mathcal{L}^{(m)}}{\partial W_{l,k,c,j}^{(i)}} W_{l,k,c,j}^{(i)} \right| \tag{9}$$

where $W_{l,k,c,j}^{(i)}$ denotes the weight element at spatial position (l,k), input channel c, and output filter j in layer i.

The gradient saliency vector $s_{i,j}^{\text{grad}}$ highlights which filters would cause the largest increase in loss if removed (those with highest $s_{i,j}^{\text{grad}}$). Conversely, filters with very low $s_{i,j}^{\text{grad}}$ contribute little to the loss and are prime candidates for pruning unless they carry other importance (which is where the frequency score comes in).

3) Fusion of Scores: Once we have both $s_{i,j}^{\text{freq}}$ and $g_{i,j}^{\text{grad}}$ for every filter j in layer i, we combine them to form the final hybrid importance score $s_{i,j}$. A straightforward combination is a weighted product or sum. We choose to multiply the gradient saliency by a transformed frequency score, as this naturally zeroes out filters that are low on either metric. Specifically, we use an element-wise multiplication with an exponent to tune the influence:

$$s_{i,j} = s_{i,j}^{\text{grad}} \odot \left(s_{i,j}^{\text{freq}} + \varepsilon \right)^{\alpha}, \tag{10}$$

where $f^{(l)}$ is the vector of frequency scores for layer l (with components $s_{l,j}^{\text{freq}}$ for each filter j in that layer), $\varepsilon = 10^{-8}$ is a small constant to avoid multiplying by zero, and α is a hyperparameter that determines how strongly the frequency score influences the hybrid score.

C. Pruning Mask Strategy and Adaptive Schedule

After computing the hybrid saliency scores $s_{i,j}$ for all filters, HSGSP proceeds to remove the least important filters in a structured way. We employ a pruning mask that indicates which filters (and corresponding channels) are to be pruned at each iteration. The mask is determined by a global threshold on the hybrid scores, combined with per-layer constraints.

1) Mask Computation: At each pruning step, all filter scores $s_{i,j}$ are combined and sorted in ascending order. According to the pruning fraction π_t, filters with the lowest scores are removed until the target count is reached, while ensuring each layer i keeps at least C_i^{min} filters to avoid over-pruning. After determining the mask, filters and their related kernels, channels, and parameters are deleted. Dependent components such as batch normalization or fully connected inputs are updated, and for simple networks like VGG, new smaller layers are created. Finally, a short fine-tuning on training data restores performance, allowing remaining weights to adapt and recover accuracy lost during pruning.

2) Adaptive Pruning Schedule: A key novelty in HSGSP lies in the self-adaptive control of the pruning rate for each iteration. An initial pruning fraction π_0 is first applied, meaning that in the initial iteration. This fraction is maintained for iterations $t = 1$ up to the designated taper point T_{taper}. Up to t^{th} iteration, a constant fraction π_0 is pruned at each step, provided that the validation loss indicates pruning may continue safely. After this stage, as the network is reduced in size and the risk of over-pruning increases, the pruning

fraction is gradually annealed to a smaller value. Specifically, for $t > T_{\text{taper}}$, π_t is linearly decreased from π_0 to a lower bound by the final iteration T. The schedule can be described as:

- Initial pruning rate: $\pi_t = \pi_0$ for $t \leq T_{\text{taper}}$
- Tapered pruning rate: for $t > T_{\text{taper}}$, let $D = \max(1, T - T_{\text{taper}})$ be the number of iterations in the tapering phase. We define a progress ratio

$$\gamma_t = min(1, \frac{t - T_{\text{taper}}}{D}) \tag{11}$$

which goes from 0 at $t = T_{\text{taper}} + 1$ to 1 at $t = T$. Then the pruning fraction is decreased linearly as:

$$\pi_t = \pi_0 - (\pi_0 - \pi_{\min})\,\gamma_t \tag{12}$$

where $\pi_{\min}$ is the minimum pruning fraction.

Beyond the preset schedule, pruning effects on accuracy are tracked through validation control. After each prune–fine-tune step, performance on a validation set D_{val} measures accuracy loss. If this drop exceeds a limit $\Delta_{\max}$, pruning halts early to avoid over-compression. In tests, the adaptive schedule plus validation check let HSGSP stop automatically once accuracy declined past tolerance, even if more iterations were possible.

IV. Experiments and Results

A. Experimental Setup

Datasets: The method is tested on CIFAR-10 and CIFAR-100 image classification benchmarks. CIFAR-10 includes 60,000 color images (32×32), divided into 10 classes with 50,000 for training and 10,000 for testing. CIFAR-100 has the same size and count but 100 classes, making it more fine-grained and difficult. These datasets represent different complexity levels, allowing evaluation of how well the method adapts to varying dataset difficulty when deciding pruning levels.

Base Model: VGG-16BN serves as the base model for pruning. This 16-layer CNN is over-parameterized for CIFAR, making it suitable for pruning tests. After full training, it achieves about 93–94% on CIFAR-10 and 73-74% Top-1 on CIFAR-100, matching reported baselines. For CIFAR-100, Top-5 accuracy is around 91%, while for CIFAR-10 it's nearly 100% due to its small class set.

Hardware All pruning experiments were conducted on a NVIDIA GeForce RTX 3090 Ti with 24GB Memory. The implementation of HSGSP (including FRN training and iterative pruning) is done in Python 3.10 using TensorFlow 2.x. While the absolute runtimes are not the focus, we note that one iteration of pruning (including scoring, mask application, and fine-tuning 25 epochs) on CIFAR-100 took roughly 4 minutes on this setup.

Each main experiment is repeated with three different random seeds and report the mean and standard deviation of the final validation accuracy. Pruning schedules and training hyperparameters are kept identical across seeds to ensure a fair comparison.

B. Implementation

We run iterative pruning for 30 iterations, pruning within each iteration until validation accuracy drops by at most $\Delta_{\max} = 0.01$, then fine-tune for 25 epochs. The initial prune fraction is $\pi_0 = 0.08$, with a per-layer constraint of at least $C_i^{\min} = 8$ filters retained. Unless stated otherwise, we use $\alpha = 0.5$ for the hybrid score and set the initial mixing factor to $\kappa_0 = 0.5$.

These values were chosen based on a few preliminary trials on CIFAR-10 to ensure the network is pruned gradually and to balance the influence of frequency and gradient criteria. The fine-tuning epochs (25) are relatively small, relying on multiple iterations rather than heavy retraining in one go.

To build the data used for computing supervision signals, an activation dataset D_a is defined as a clean subset of the training split (after the validation split), i.e., we use the training portion without data augmentation or mixup. For CIFAR with 50k training images and a 10% validation split, D_a contains approximately 45k images. In our implementation, the FRN builder reads up to 256 batches (batch size 256), which is sufficient to cover all images in D_a for CIFAR. FRN samples are constructed per convolutional filter by aggregating statistics over images in D_a, yielding one training sample per filter across convolutional layers (4,224 samples for VGG-style networks). The 3-64-32-3 MLP FRN was trained for 15 epochs on the filter dataset using Adam optimizer with a 0.001 learning rate.

The experiments fix the optimizer, learning-rate schedule, number of pruning iterations, and the number of fine-tuning epochs per iteration to ensure a controlled comparison. The remaining hyperparameters relate to pruning control: the fusion coefficient α, the adaptive schedule ratio κ, and the per-iteration prune fraction. These pruning-related parameters are selected via a small set of pilot runs using the validation split and are then kept fixed for all reported results and method comparisons.

C. Results on CIFAR-10

Table I summarizes the pruning results on CIFAR-10 with VGG16-BN. Since each work trains its base model differently, resulting in varying baseline accuracies, accuracy drop was used as the metric. It measures the difference in accuracy between the pruned and original models, where a negative value indicates the pruned model performs better. For each method, the number of remaining parameters and FLOPs after pruning were also reported. Two operating points of HSGSP were presented. HSGSP achieves significant model compression while maintaining competitive or even improved accuracy. In the first configuration, where accuracy preservation is prioritized, the model retains 1.81M parameters and 282.91M FLOPs, corresponding to an approximately 90% reduction in parameters relative to the unpruned network, while improving accuracy by 0.78% over baseline. In the second configuration, where higher compression is targeted, the model is pruned to 0.90M parameters and 128.52M FLOPs—a 94% reduction—with only a 0.58% accuracy loss.

TABLE I
PRUNING RESULTS ON CIFAR-10 WITH VGG-16BN

Method	Base Acc. (%)	Pruned Acc. (%)	Acc. Drop (%)	Params (M)	FLOPs (M)
CFDP Pruned [17]	93.96	94.10	-0.14	2.76	131.17
APRS [24]	93.87	94.73	-0.86	2.92	109.09
Novel Deep-Learning [25]	94.17	93.22	0.95	1.10	185.80
SPECTRAL [26]	93.51	93.90	-0.39	5.26	211.62
NUCLEAR [26]	93.51	93.81	-0.30	5.26	211.62
FROBENIUS [26]	93.51	93.82	-0.31	5.26	211.62
NV-it [27]	85.05	84.83	0.22	7.60	338.04
GFI-AP [28]	92.54	92.09	0.45	3.20	177.00
HSGSP (Ours)	**93.64**	**94.42±0.12**	**-0.78±0.12**	**1.81**	**282.91**
HSGSP (Ours)	**93.64**	**93.06±0.10**	**0.58±0.10**	**0.90**	**128.52**

Compared with recent structured pruning baselines, HSGSP delivers a superior balance between compactness and performance. Although APRS [24] and CFDP [17] achieve slightly higher FLOPs reduction (approximately 110M – 130M FLOPs), they require more parameters (approximately 2.7M – 2.9M) and occasionally degrade accuracy. SPECTRAL, NUCLEAR, and FROBENIUS [[26] slightly improve model accuracy (about 0.4%) but compress the model only to 5.26M parameters—less than half the reduction achieved by HSGSP. NV-it [27] and GFI-AP [28] further increase FLOPs but remain less parameter-efficient.

These results confirm that the proposed hybrid spectral–gradient criterion prunes filters more selectively, effectively removing redundancy while retaining generalizable low-frequency structures. The slight accuracy gain at moderate compression suggests a regularization effect, whereas deeper pruning still preserves acceptable accuracy due to the validation-aware adaptive schedule.

To complement parameters and FLOPs, we benchmark inference latency on a Raspberry Pi 5 (4GB) using TensorFlow 2.10 with FP32 `.tflite` models. We report mean batch=1 inference latency over 1,000 runs after 50 warm-up runs (inference-only). As shown in Table II, HSGSP reduces complexity by about 90% and achieves a 3.45× speedup in the first configuration and up to 7.54× with less than 1% accuracy drop; notably, the first configuration slightly improves CIFAR-10 top-1 accuracy (93.78% → 94.65%), confirming practical efficiency gains on-device.

TABLE II
EVALUATION ON EDGE DEVICE

Model	Params (M)	FLOPS (M)	Model size (MB)	Top-1 Acc. (%)	Latency (ms/image)
Baseline	15.00M	627.20	57.23	93.78	15.16
Pruned Model 1	1.81M	282.91	6.94	94.65±0.11	4.39
Pruned Model 2	0.90M	128.53	3.45	93.37±0.12	2.01

D. Results on CIFAR-100

The pruning results on the more challenging CIFAR-100 dataset are next examined, as summarized in Table III. Since CIFAR-100 contains 100 classes, it is considered significantly more complex; therefore, maintaining accuracy after pruning is expected to be more difficult, making the advantage of an adaptive strategy such as HSGSP more apparent.

HSGSP is presented again in two settings: one in which a slight accuracy increase is observed (a –0.25% drop, corresponding to a +0.25% gain in accuracy) and another in which prioritize model size (with a small accuracy drop of 0.42%). These settings correspond to different pruned model sizes. The model with a –0.25% drop retains approximately 4.94M parameters and 354.23M FLOPs, while the model with a 0.42% drop retains 3.39M parameters and 231.77M FLOPs. For comparison, the unpruned VGG-16 model is characterized by 15.0M parameters and roughly 627.5M FLOPs on CIFAR-100.

In comparison, the methods from Sun and Shi [26] (SPECTRAL, NUCLEAR, FROBENIUS) pruned VGG-16 to 5.29M parameters (and 211.6M FLOPs) and reported very small accuracy drops (0.17% to 0.64%). Although HSGSP has a slightly lower compression rates: it removes about 63.8% of the parameters, versus 64% removal by [26], it moderately improves the accuracy. This is a remarkable result, indicating HSGSP's hybrid criteria can identify far more redundancy without harming performance. The likely reason is that [26]'s criteria (based on norms) are more conservative in removing filters, whereas HSGSP, guided by both spectral and gradient signals and constrained by validation feedback, can safely prune the model.

TABLE III
PRUNING RESULTS ON CIFAR-100 WITH VGG-16BN

Method	Base Acc. (%)	Pruned Acc. (%)	Acc. Drop (%)	Params (M)	FLOPs (M)
SPECTRAL [26]	73.53	73.36	0.17	5.29	211.64
NUCLEAR [26]	73.53	73.03	0.50	5.29	211.64
FROBENIUS [26]	73.53	72.89	0.64	5.29	211.64
GFI-AP [28]	73.97	73.68	0.30	6.5	247
HSGSP (Ours)	73.22	**73.74±0.15**	**-0.25±0.15**	**4.94**	**354.23**
HSGSP (Ours)	73.22	**72.80±0.13**	**0.42±0.13**	**3.39**	**231.77**

E. Results on Tiny-ImageNet

To address concerns regarding the low image resolution of CIFAR datasets, we additionally evaluate HSGSP on Tiny-ImageNet with 64×64 inputs under the same iterative prune and fine-tune protocol. We report top-1 validation accuracy along with model complexity metrics (parameters and FLOPs) across pruning iterations.

Table IV reports representative checkpoints on Tiny-ImageNet (64×64) to compare the pruning trajectories of the gradient-only criterion and the proposed hybrid baseline. Starting from the same unpruned model, both configurations retain comparable accuracy at early iterations while reducing computation substantially. Importantly, in the moderate-compression regime, the hybrid criterion reaches an ≈2% accuracy-drop checkpoint with stronger efficiency: it reduces the model to 9.89M parameters and 875.60M FLOPs at 60.03% accuracy, whereas the gradient-only counterpart attains a similar accuracy level at a larger compute cost which is 12.88M parameters and 1064.24M FLOPs at 59.87%. This

indicates that incorporating frequency-aware information can preserve task-relevant filters under pruning, enabling greater compute reduction at comparable accuracy.

TABLE IV
PRUNING RESULTS ON TINY IMAGENET WITH VGG-16BN

Model	Iteration	Acc. (%)	Params (M)	FLOPs (M)
Baseline	–	61.88	15.89	2508.98
Gradient-only	1	61.14	15.16	1684.30
Gradient-only (≈2% drop)	3	59.87	12.88	1064.24
Hybrid (best acc.)	1	61.98	15.06	1783.70
Hybrid (≈2% drop)	5	60.03	9.89	875.60

TABLE V
ABLATION OF FUSION COEFFICIENT α

Budget	α	Iter.	Params (M)	FLOPs (M)	Acc. (%)
≈ 1.85M	GRAD. ONLY	16	1.85	275.36	94.06
	0.25	17	1.78	273.64	94.18
	0.50	**17**	**1.819**	**282.92**	**94.42**
	1.00	17	1.87	266.28	94.40
	FREQ. ONLY	23	1.82	58.98	82.41
≈ 0.85M	GRAD. ONLY	29	0.833	122.70	93.37
	0.25	**30**	**0.837**	**120.36**	**93.40**
	0.50	30	0.858	120.40	93.32
	1.00	30	0.874	114.09	92.84
	FREQ. ONLY	30	1.264	38.53	81.46

F. Ablation Studies

We conduct an ablation study on CIFAR-10 to quantify the role of the fusion coefficient α in the proposed importance score (Eq.(10)). Across all settings, we keep the iterative pruning schedule and fine-tuning protocol fixed and vary only α to control the contribution of the frequency-based score relative to the gradient-based score: $\alpha = 0$ disables the frequency term (gradient-only), while $\alpha > 0$ progressively increases the influence of the frequency component; we also evaluate a frequency-only variant that ranks filters using only the frequency score to test whether spectral cues alone suffice. As shown in TableV, hybrid settings consistently outperform the gradient-only baseline at matched budgets, with $\alpha = 0.5$ achieving the best accuracy at ≈1.82M parameters (94.42%) and $\alpha = 0.25$ performing best near 0.85M parameters (93.40%). In contrast, the frequency-only variant suffers a large degradation (82.41% at ≈1.82M and 81.46% at ≈1.26M parameters), indicating that spectral information alone is not sufficiently task-specific and that the strongest accuracy–efficiency trade-off comes from combining frequency-aware cues with gradient saliency.

G. Discussion

Fig. 2 visualizes the evolution of accuracy over 30 pruning iterations under different fusion settings, together with the parameter trajectory (green bars) of the reference run ($\alpha = 0$). The overall trend is consistent across settings: as pruning

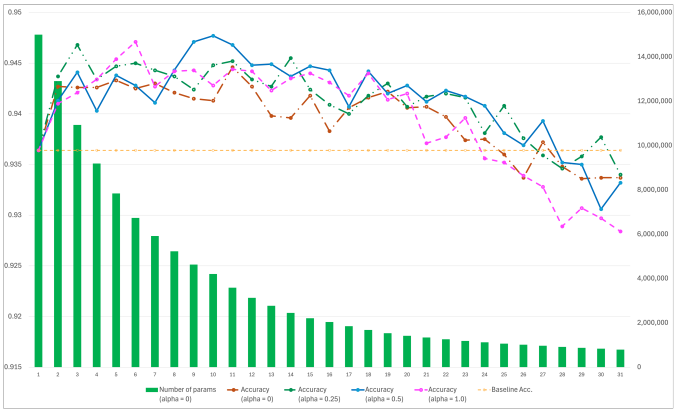

Fig. 2. Evolution of validation accuracy and parameter count across 30 pruning iterations under varying fusion coefficients α

progresses and the model becomes progressively smaller, accuracy initially improves slightly and then gradually declines under more aggressive compression (late iterations). Compared with the gradient-only baseline ($\alpha = 0$), hybrid configurations with moderate frequency contribution, notably $\alpha = 0.25$ and $\alpha = 0.5$, maintain higher and more stable accuracy over most iterations, indicating that frequency-aware cues help preserve structurally important filters during repeated prune and fine-tune cycles. In particular, $\alpha = 0.5$ achieves the strongest performance in the mid-compression regime (approximately iterations 8 - 20), suggesting that balancing gradient saliency and frequency information yields the best accuracy and efficiency trade-off. In contrast, the $\alpha = 1.0$ curve exhibits a visibly larger degradation in the heavily pruned regime (late iterations), implying that excessively emphasizing frequency-aware scoring reduces robustness when the network capacity becomes limited.

The figure also suggests that the adaptive pruning schedule promotes a gradual compression path. The parameter count decreases smoothly across iterations rather than collapsing in a few aggressive steps, which helps fine-tuning recover accuracy after each pruning action and improves stability compared with fixed-ratio schedules. Consistently, the accuracy curves exhibit a relatively long stable plateau in the early to mid iterations, where performance remains close to (or above) the baseline despite continuous compression, indicating that the controller prunes conservatively when the model is most sensitive. In late iterations, accuracy degrades more noticeably across all α settings, revealing a capacity-limited regime with diminishing returns, which motivates the adaptive design as a mechanism to dynamically modulate (or potentially slow/stop) pruning based on validation feedback to delay severe performance drops.

To address how the frequency component affects pruning decisions beyond gradients alone, Fig. 3 visualizes the joint distribution of filter scores at a representative pruning step (decision iteration = 12). Each point corresponds to one filter, positioned by its gradient saliency (x-axis, log-scale) and its frequency score (y-axis); green points are kept and red points are pruned.

Under the gradient-only setting, pruning follows an almost vertical threshold on gradient saliency, which can discard

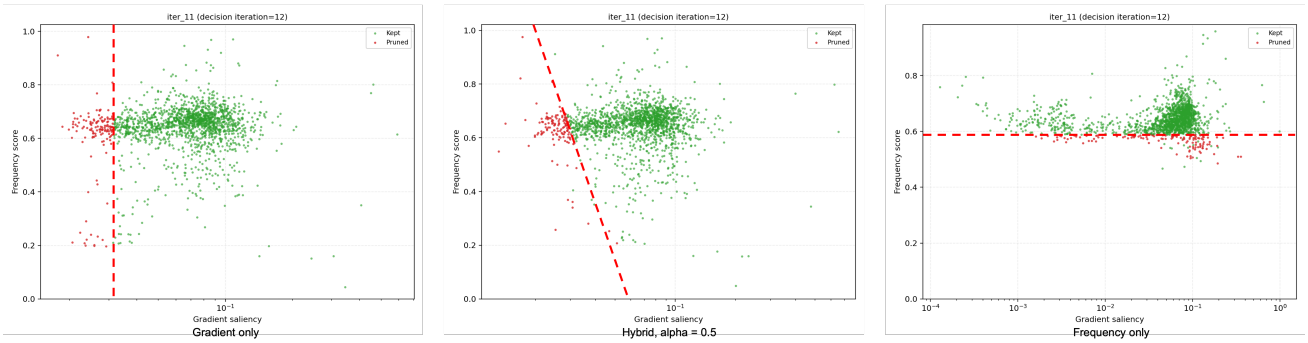

Fig. 3. Joint distribution of filter scores and pruning decisions at iteration 12 across gradient-only, hybrid ($\alpha = 0.5$), and frequency-only configurations

filters with weak gradients despite strong frequency scores. Frequency-only yields an approximately horizontal separation and ignores gradients, explaining its inferior ablation accuracy: spectral cues alone are not sufficiently task-specific. In contrast, the hybrid setting with $\alpha = 0.5$ produces an oblique boundary that preserves moderately salient filters when supported by high frequency scores, using spectral information as an auxiliary signal to complement instantaneous gradients.

Limitations: HSGSP has so far been tested only on small datasets and one model. Evaluating it on larger sets like ImageNet and newer architectures (ResNets, Transformers) is key to assess scalability. FRN should generalize since DCT/FFT apply to any conv filter, though larger models may need tuning of B or FRN size. FRN scoring adds some overhead, negligible for small models but notable for big ones, though inference stays fast and pruning can overlap with training.

Structured pruning is often combined with knowledge distillation (KD) [29] to improve accuracy recovery under aggressive compression, where a pruned student is trained to match a stronger teacher model. While our study focuses on isolating the effect of the proposed spectral - gradient pruning criterion within an iterative pipeline, KD is complementary and can be incorporated straightforwardly by adding a distillation loss (e.g., logits- or feature-based matching) during the fine-tuning stage after each pruning iteration. Investigating the interaction between HSGSP and KD, especially under stronger compression regimes, is an interesting direction for future work.

V. Conclusion

This paper introduces Hybrid Spectral–Gradient Saliency Pruning (HSGSP), a new framework for structured CNN pruning that merges frequency-domain analysis with gradient-based importance measures. It tackles two gaps: missing dataset-aware pruning rates and limited use of spectral cues. A lightweight Frequency Relevance Network (FRN) learns to predict filter importance from weight-domain spectral descriptors, reducing reliance on gradient-only scoring during pruning iterations. HSGSP fuses FRN scores with Taylor gradient saliency into a hybrid metric, ensuring filters pruned are redundant both spectrally and task-wise. An adaptive validation-driven schedule automatically tunes pruning intensity to maintain accuracy.

Acknowledgments

Some parts in implementation source code are done by OpenAI Codex.

This research is funded by Vietnam National University Ho Chi Minh City (VNU-HCM) under grant number DS2024-26-03.

References

[1] Y. He, P. Liu, Z. Wang, Z. Hu, and 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 (CVPR)*, 2019, pp. 4340–4349, doi: https://doi.org/10.1109/CVPR.2019.00447.

[2] M. Lin, R. Ji, Y. Wang, Y. Zhang, B. Zhang, Y. Tian, and L. Shao, "HRank: Filter pruning using high-rank feature map," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 1529–1538, doi: https://doi.ieeecomputersociety.org/10.1109/CVPR42600.2020.00160.

[3] X. Ding, T. Hao, J. Tan, J. Liu, J. Han, Y. Guo, and G. Ding, "Resrep: Lossless CNN pruning via decoupling remembering and forgetting," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, Oct 2021, pp. 4510–4520, doi: https://doi.ieeecomputersociety.org/10.1109/ICCV48922.2021.00447.

[4] Y. Chen, R. Zhou, B. Guo, Y. Shen, W. Wang, X. Wen, and X. Suo, "Discrete cosine transform for filter pruning," *Applied Intelligence*, vol. 53, no. 3, pp. 3398–3414, 2023, doi: https://doi.org/10.1007/s10489-022-03604-2.

[5] S. Zhang, M. Gao, Q. Ni, and J. Han, "Filter pruning with uniqueness mechanism in the frequency domain for efficient neural networks," *Neurocomputing*, vol. 530, pp. 116–124, 2023, doi: https://doi.org/10.1016/j.neucom.2023.02.004.

[6] Y. He, G. Kang, X. Dong, Y. Fu, and Y. Yang, "Soft filter pruning for accelerating deep convolutional neural networks," in *arXiv preprint arXiv:1808.06866*, 2018, doi: https://doi.org/10.24963/ijcai.2018/309.

[7] S. Lin, R. Ji, C. Yan, B. Zhang, L. Cao, Q. Ye, F. Huang, and D. Doermann, "Towards optimal structured CNN pruning via generative adversarial learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 2790–2799, doi: https://doi.ieeecomputersociety.org/10.1109/CVPR.2019.00290.

[8] Y. Tang, Y. Wang, Y. Xu, D. Tao, C. Xu, C. Xu, and C. Xu, "SCOP: Scientific control for reliable neural network pruning," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2020, doi: https://doi.org/10.48550/arXiv.2010.10732.

[9] Y. Wang, X. Zhang, L. Xie, J. Zhou, H. Su, B. Zhang, and X. Hu, "Pruning from scratch," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, 2020, pp. 12 273–12 280, doi: https://doi.org/10.1609/aaai.v34i07.6910.

[10] S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding," *arXiv preprint arXiv:1510.00149*, 2015, doi: 10.48550/arXiv.1510.00149.

[11] Y. Zhang, H. Bai, H. Lin, J. Zhao, L. Hou, and C. V. Cannistraci, "Plug-and-play: An efficient post-training pruning method for large language models," in *The Twelfth International Conference on Learning Representations*, 2024, doi: 10.20944/preprints202310.1487.v2.

[12] E. Frantar and D. Alistarh, "Optimal brain compression: A framework for accurate post-training quantization and pruning," *Advances in Neural Information Processing Systems*, vol. 35, pp. 4475–4488, 2022, doi: 10.48550/arXiv.2208.11580.

[13] W. Kwon, S. Kim, M. W. Mahoney, J. Hassoun, K. Keutzer, and A. Gholami, "A fast post-training pruning framework for transformers," *Advances in Neural Information Processing Systems*, vol. 35, pp. 24 101–24 116, 2022, doi: 10.48550/arXiv.2204.09656.

[14] Y. Li, R. Gong, X. Tan, Y. Yang, P. Hu, Q. Zhang, F. Yu, W. Wang, and S. Gu, "Brecq: Pushing the limit of post-training quantization by block reconstruction," *arXiv preprint arXiv:2102.05426*, 2021, doi: 10.48550/arXiv.2102.05426.

[15] G. Xiao, J. Lin, M. Seznec, H. Wu, J. Demouth, and S. Han, "Smoothquant: Accurate and efficient post-training quantization for large language models," in *International Conference on Machine Learning*. PMLR, 2023, pp. 38 087–38 099. [Online]. Available: https://proceedings.mlr.press/v202/xiao23c.html

[16] Z. Liu, J. Xu, X. Peng, and R. Xiong, "Frequency-domain dynamic pruning for convolutional neural networks," *Advances in neural information processing systems*, vol. 31, 2018. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2018/hash/a9a6653e48976138166de32772b1bf40-Abstract.html

[17] S. Khaki and W. Luo, "CFDP: Common frequency domain pruning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2023, pp. 4715–4724, doi: https://doi.org/10.1109/CVPRW59228.2023.00499.

[18] T. Dettmers and L. Zettlemoyer, "Sparse networks from scratch: Faster training without losing performance," *arXiv preprint arXiv:1907.04840*, 2019, doi: https://doi.org/10.48550/arXiv.1907.04840.

[19] A. I. Nowak, Ł. Gniecki, F. Szatkowski, and J. Tabor, "Sparser, better, deeper, stronger: Improving sparse training with exact orthogonal initialization," *arXiv preprint arXiv:2406.01755*, 2024, doi: https://doi.org/10.48550/arXiv.2406.01755.

[20] D. C. Mocanu, E. Mocanu, T. Pinto, S. Curci, P. H. Nguyen, M. Gibescu, D. Ernst, and Z. A. Vale, "Sparse training theory for scalable and efficient agents," *arXiv preprint arXiv:2103.01636*, 2021, doi: https://doi.org/10.48550/arXiv.2103.01636.

[21] S. Liu, I. Ni'mah, V. Menkovski, D. C. Mocanu, and M. Pechenizkiy, "Efficient and effective training of sparse recurrent neural networks," *Neural Computing and Applications*, vol. 33, pp. 9625–9636, 2021, doi: https://doi.org/10.1007/s00521-021-05727-y.

[22] S. Liu, Y. Tian, T. Chen, and L. Shen, "Don't be so dense: Sparse-to-sparse gan training without sacrificing performance," *International Journal of Computer Vision*, vol. 131, no. 10, pp. 2635–2648, 2023, doi: https://doi.org/10.1007/s11263-023-01824-8.

[23] S. Liu, T. Chen, X. Chen, L. Shen, D. C. Mocanu, Z. Wang, and M. Pechenizkiy, "The unreasonable effectiveness of random pruning: Return of the most naive baseline for sparse training," *arXiv preprint arXiv:2202.02643*, 2022, doi: https://doi.org/10.48550/arXiv.2202.02643.

[24] Q. Sun, S. Cao, and Z. Chen, "Filter pruning via automatic pruning rate search," in *Proceedings of the Asian conference on computer vision*, 2022, pp. 4293–4309, doi: https://doi.org/10.1007/978-3-031-26351-4_36.

[25] M. Zhao, X. Tong, W. Wu, Z. Wang, B. Zhou, and X. Huang, "A novel deep-learning model compression based on filter-stripe group pruning and its iot application," *Sensors*, vol. 22, no. 15, p. 5623, 2022, doi: https://doi.org/10.3390/s22155623.

[26] X. Sun and H. Shi, "Towards better structured pruning saliency by reorganizing convolution," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2024, pp. 2204–2214, doi: https://doi.ieeecomputersociety.org/10.1109/WACV57701.2024.00220.

[27] C. I. López-González, E. Gascó, F. Barrientos-Espillco, E. Besada-Portas, and G. Pajares, "Filter pruning for convolutional neural networks in semantic image segmentation," *Neural Networks*, vol. 169, pp. 713–732, 2024, doi: https://doi.org/10.1016/j.neunet.2023.11.010.

[28] B. D. Milton, S. D. Roy, P. Singh, B. Lall, and S. D. Joshi, "Adaptive cnn filter pruning using global importance metric," *Comput Vis Image Understanding*, vol. 222, p. 103511, 2022, doi: https://doi.org/10.1016/j.cviu.2022.103511.

[29] K. Do, T. H. Le, D. Nguyen, D. Nguyen, H. Harikumar, T. Tran, S. Rana, and S. Venkatesh, "Momentum adversarial distillation: Handling large distribution shifts in data-free knowledge distillation," *Advances in neural information processing systems*, vol. 35, pp. 10 055–10 067, 2022.

Thanh-Thien Nguyen received B.S. degree in Information Technology in 2013 and M.Sc. degree in Computer Science in 2018 from the University of Science, Vietnam National University Ho Chi Minh City. He is currently pursuing a Ph.D. degree in Computer Science at University of Information Technology, Vietnam National University Ho Chi Minh City, with a focus on efficient deep learning. His research interests include machine learning, computer vision and their applications.

Hoang-Loc Tran received his B.S. degree in Computer Engineering in 2018 and M.Sc. degree in Computer Science from the University of Information Technology, Vietnam National University Ho Chi Minh City. He has been actively engaged in research and teaching in the areas of Artificial Intelligence on Embedded Systems, AI model optimization, and embedded vision. His current research focuses on developing efficient and adaptive deep learning models suitable for resource-constrained hardware platforms, integrating TinyML approaches to bridge artificial intelligence and embedded technologies.

Vo-Chi-Dung Nguyen is currently pursuing a degree in Computer Science from the University of Information Technology, Vietnam National University. He specializes in research areas including computer vision, model optimization, and deep learning algorithms. His current research centers on improving machine learning models and uncertainty quantification in Explainable AI to enhance the reliability and transparency of intelligent systems, advancing trustworthy and interpretable machine learning.

Viet-An Nguyen is a Computer Science student currently undergoing collaborative training between the University of Information Technology (UIT), Vietnam, and Birmingham City University (BCU), United Kingdom. His preferred research directions include transfer learning methods for models on resource-constrained environments and accelerating the training of neural network architecture-based applications. His current research focuses on training Vietnamese language classification models using limited Vietnamese datasets, leveraging data augmentation and transfer learning.

Duc-Lung Vu received B.S. and M.Sc. degrees in Computer Engineering from the Peter the Great St.Petersburg Polytechnic University in 1998 and 2000, respectively. He got his Ph.D. in Computer Science from Saint Petersburg Electrotechnical University in 2006. He has been working at the University of Information Technology, Vietnam National University Ho Chi Minh City, as an Associate Professor since 2015 and Chancellor of the school since 2020. His research interests include machine learning, human-computer interaction, embedded systems and digital system design on FPGA.