FMIP: Feature Map Importance Pruning for Efficient CNN Optimization

Ali Muhammad Shaikh
University of Science and Technology of China
Hefei, China
alims@mail.ustc.edu.cn

Yun-bo Zhao*
University of Science and Technology of China
Hefei, China
ybzhao@ustc.edu.cn

Yu Kang
University of Science and Technology of China
Hefei, China
kangduyu@ustc.edu.cn

Aakash Kumar
Zhongshan Institute of Changchun University of Science and Technology, China
Zhonhshan, China
aakash@cust.edu.cn

***Abstract*— Deep convolutional neural networks (CNNs) have reformed numerous fields, comprising computer vision and natural language processing. Nevertheless, CNNs' computational complexity frequently raises obstacles to utilization on resource-constrained devices. Different optimization strategies have been proposed in response to this issue, comprising quantization, knowledge distillation, and pruning. This paper familiarizes a novel pruning method, Feature Map Importance Pruning (FMIP), designed to optimize the performance of deep CNNs while reducing computational costs. The FMIP computes the importance of feature maps within convolutional layers according to the area of their activation values, facilitating a systematic approach to pruning decisions. By detecting and eliminating the redundant feature maps, FMIP can considerably cut the number of parameters and FLOPs in a CNN model without conceding accuracy. This is chiefly advantageous for using CNNs on devices with limited memory and computational power.**

We evaluated FMIP on various CNN architectures, including VGG16 and ResNet50, using datasets such as CIFAR-10 and ImageNet. Our experiments demonstrated significant model compression while preserving high accuracy. Specifically, FMIP achieved the following results: VGG16 with CIFAR-10 demonstrated an 81.9% reduction in parameters and a 48.6% reduction in FLOPs, with an accuracy of 93.33%. ResNet50 with ImageNet achieved an 83.11% reduction in parameters and a 71.55% reduction in FLOPs, with an accuracy of 92.03%.

Keywords*— *Deep CNNs, Feature Map Importance, Model compression, Filter Pruning, Optimization

1 Introduction

To enhance the efficiency of Convolutional Neural Networks (CNNs), researchers have utilized advanced techniques across various machine learning domains, such as object detection [1], [2], fault detection in UAVs [3], and image recognition [4], [5]. However, achieving outstanding results also poses significant challenges. These challenges include complex architectures that are suboptimal regarding memory usage and require substantial computational resources, particularly for embedded and mobile devices. Additionally, these architectures incur considerable costs during inference. Consequently, model compression has garnered significant attention from researchers as a means to address the size issue of CNN architectures. Even so, CNNs are inherently computationally intensive, leading to a substantial increase in floating-point operations (FLOPS) [6] due to the extensive trainable parameters and convolution operations involved.

In recent years, researchers have proposed various model compression and acceleration methods [6], [7], [8]. Based on pruning granularity, network pruning can be categorized into unstructured pruning [9] and structured pruning [10]. A common form of unstructured pruning is weight pruning, which reduces network parameters by eliminating insignificant weights in the filters. Unstructured pruning requires specialized software or hardware for model acceleration, whereas structured pruning does not face this issue. Consequently, structured pruning has garnered more attention in recent years. Structured pruning primarily involves filter pruning.

The key to filter pruning lies in selecting which filters to remove. Our proposed pruning strategy FMIP (Feature Map Importance Pruning) is based on the feature map area, targeting layers and ranking feature maps according to "area" in terms of the magnitude of activations. This approach provides a clear, interpretable metric to evaluate the importance of each filter in the network. The underlying assumption is that feature maps with smaller activations (areas) contribute less to the overall task and can be safely removed, thereby reducing the model's complexity. Nevertheless, we rank the feature maps by their scores, which reflect how "active" or "informative" (see Figure 1). This approach is computationally efficient because calculating the area involves only basic summations, making it practical even for large models. Moreover, the combination of iterative pruning and fine-tuning enables the network to gradually adapt to the removal of less significant filters, thereby minimizing the performance loss that often occurs with more aggressive pruning methods.

2 Literature Review

The primary objective of filter pruning is to assess the significance of filters and eliminate those considered unimportant [11], [12]. Consequently, re-training is necessary

after each pruning step to compensate for the reduction in accuracy. In reference [13], the importance of filters was evaluated using a portion of the training data based on the output feature map. In [14], pruning was performed using a greedy technique that determined filter importance by evaluating the model's accuracy post-pruning. Additionally, in [15], feature maps, or pruning activations, were employed to create faster CNNs. This method involves removing filters from specific input locations while often retaining them in other locations, resulting in minimal overall filter compression. This approach may also be viewed as excluding filters at specific input locations, but these filters are often retained at other locations, leading to limited filter compression.

In [16] employs high-rank feature map selection for filter pruning. HRank ranks feature maps based on their information content using matrix rank, ensuring that the most significant feature maps are preserved during pruning. Another method named SCSP (Spectral Clustering Filter Pruning) was introduced in [17], which employs a self-adaptive method to prune filters, concentrating on the spectral characteristics of filters. Our proposed strategy "Feature Map Importance Pruning (FMIP) differs from these methods. Firstly, it is more geometrically motivated by spatial coverage, offering a more straightforward metric that can be easily interpreted in terms of input space activation. Secondly, it is based on the area under feature map activations, concentrating on measuring the spatial activation area rather than rank, providing it with a unique advantage in understanding the feature map's contribution through the coverage area. Furthermore, this method is less computationally intensive than spectral clustering or HRank techniques, making it simpler to implement and potentially quicker to execute without extensive fine-tuning.

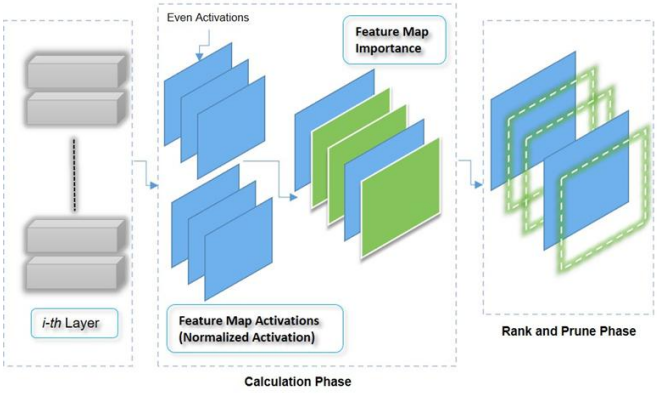

Figure 1 Feature Map Importance Pruning (FMIP) Process

3 Pruning filters via Rank and Importance

To optimize (CNNs) through weighing and pruning feature maps characterized by their importance scores, which can be conceptually associated with their active "area" of contribution across each layer. we set off with the activation values computation from the convolutional layers of the CNN model.

3.1 Ranking and Importance Score Calculation

For a specific convolutional layer n, let X_n signify the input to the layer, and $X_n \in \mathbb{R}^{C_{\text{in}} \times H \times W}$ (with C_{in} is denoted as the number of input channels, H implies the height, and W is the width of the input). The convolution operation provides feature maps L_n as output, where:

$$L_n = W_n * X_n + b_n \tag{1}$$

Now, W_n are denoted as convolutional filters of layer n alongside dimensions $C_{out} \times C_{in} \times k_h \times k_w$ and b_n denotes the bias. The feature map activations $L_n \in \mathbb{R}^{C_{\text{out}} \times H' \times W'}$ are then acquired after implementing the convolutional filters, where C_{out} is the number of output channels, and H', W' are the height and width of the output feature maps. The importance of feature maps is calculated by determining the spatial area covered by the activations in each feature map. The area is decided through summing the absolute activations to indicate its contribution to the forward pass. The area A_i of feature map i can be calculated as follows:

$$A_i = \sum_{h=1}^{H'} \sum_{w=1}^{W'} |A_{i,h,w}| \tag{2}$$

Where $A_{i,h,w}$ means the value at position (h, w) in the i-th feature map of a certain layer. $\sum_{h=1}^{H'} \sum_{w=1}^{W'}$ stands for the raw area score, determining the contribution of feature map i. We apply Min-Max Normalization to the areas A_i before ranking the feature maps for pruning. To scale the values between 0 and 1, we ensure that the importance scores are uniformly scaled throughout all feature maps. The normalized importance score I_i is calculated as:

$$I_i = \frac{A_i - \min(A)}{\max(A) - \min(A)} \tag{3}$$

Where $\min(A)$ and $\max(A)$ signify the minimum and maximum area values, respectively throughout all feature maps. After normalization, the feature maps are ranked by their normalized importance values. For example, Lower importance scores imply that the equivalent feature map offers less to the general activations and therefore can be intended for pruning. Let the ranking be denoted via an ordered set R such that:

$$R = \text{sort}\ (I_1, I_2, \ldots, I_n) \tag{4}$$

where n denoted as the total number of feature maps, the sort function organizes the feature maps via their normalized importance scores.

3.2 Pruning Criterion: Feature Map Area

We set a threshold τ is used to prune a definite percentage of feature maps. For instance, if $\tau = 0.3$, the bottom 30% of feature maps according to their importance scores are pruned.

$$\text{Prune } F_i \text{ if } I_i < \tau \quad (5)$$

where F_i signifies the i-th feature map. The feature maps that contribute the least to the network's activations are eliminated, successfully dropping the number of filters. where τ is the threshold value defined by the pruning rate r. Pruning a feature map F_i requires the deletion of the connected filter W_i. In the updated network, the pruned filters are efficiently neglected. We then fine-tune the model to regulate the deficit of the pruned feature maps. The final importance score, which merges both the Min-Max Normalization and area-based metric, can be defined as:

$$I_i = \frac{\sum_{h=1}^{H'} \sum_{w=1}^{W'} |A_{i,h,w}| - \min(A)}{\max(A) - \min(A)} \quad (6)$$

Equation (6) is used for ranking and pruning the feature maps in FMIP.

4 Experiments

4.1 Ranking and Importance Score Calculation

We utilized the LeNet model on the MNIST dataset as a prototype. To further demonstrate the efficiency of our proposed approach in shrinking model size, we conducted experiments on mainstream CNN architectures like VGG16 using the CIFAR10 and CIFAR100 datasets. We evaluated the model's performance post-pruning by monitoring accuracy, number of parameters, and required Floating Point Operations (FLOPs). The effectiveness of FMIP was assessed by comparing the performance before and after the pruning process, ensuring that the model maintained its predictive capability while achieving reduced complexity. Through this systematic and iterative approach, FMIP effectively ranked feature map areas and conducted a tailored pruning procedure that enhanced CNN efficiency while preserving the essential performance characteristics.

4.2 Hyperparameters Settings

All the experiments were conducted using the PyTorch framework on an NVIDIA Tesla GPU, with the execution environment Google Colab Pro. For the VGG16 model on CIFAR-10 and ResNet50 on ImagNet, the settings incorporate a layer-wise learning rate adjustment to fine-tune the learning dynamics for each layer based on its position in the network. In this setup, the learning rate for the initial convolutional layers is set to a lower value, typically 0.001, as these layers are responsible for learning the fundamental visual features. For the middle layers, the learning rate is kept slightly higher at around 0.005. The deeper layers, particularly the fully connected layers, require more aggressive learning to adapt to the classification task. Hence, the learning rate for these layers is set at the base value of 0.01.

Unlike the LeNet-5 model on MNIST, a lower learning rate of 0.001 as the task is simpler because of grayscale input images. We used Stochastic Gradient Descent (SGD) with a momentum value of 0.9 to provide stable updates during training for every model. The training is set for 100 epochs with a batch size of 128 for both ResNet50 and VGG16, but for LeNet-5 batch size is set to 64. To avert the overfitting issue, a weight decay (L2 regularization) of 1e-4 for LeNet-5 and 5e-4 is applied for the ResNet50 and VGG16 models, respectively. The model also applies data augmentation such as random cropping and horizontal flipping to boost simplification.

5 Results

In our prototype study, we applied the Feature Map Importance Pruning (FMIP) method to the LeNet network using the MNIST dataset. This approach led to a significant reduction of 77% in network parameters and a 76.5% saving in FLOPs. Impressively, despite this substantial pruning, the model maintained an accuracy of 98.86%, which is only marginally below the baseline accuracy of 99.21%.

Furthermore, we assign a pruning rate of 0.3 to the VGG16 network and perform iterative pruning. Our proposed FMIP (Feature Map Importance Pruning) approach demonstrates superior performance compared to existing filter pruning methods. Table 1 shows the results for the VGG network, we adopted the 16-layer model (comprising 13 convolutional and 3 fully connected layers) to work with the CIFAR-10 dataset. Despite pruning 81.9% of the parameters and saving 48.6% of FLOPs, we maintained an accuracy of 93.33%, only slightly above our baseline accuracy. These optimizations significantly enhance the VGG model's capability, a popular backbone for object detection and semantic segmentation, to be efficiently deployed on mobile devices.

TABLE I. TABLE 1VGG16 PRUNING PERFORMANCE ON CIFAR10

	VGG-16 on CIFAR-10 datasets (Prune results)			
Approach	Baseline (%)	FLOPs Saved (%)	Pruned (%)	Accuracy (%)
[18]	93.73	52.4	89.7	93.82
[19]	93.25	39.1	73.3	93.18
[20]	-	41.6	73.8	93.02
FMIP	**93.30**	**48.6**	**81.9**	**93.33**
[21]	-	42.5	82.2	90.73

For the ResNet50 network using the ImageNet dataset. This approach resulted in the pruning of 83.11% of the network parameters, concurrently achieving a 71.55% reduction in FLOPs. Despite these significant modifications, the model retained a commendable accuracy of 92.03%, marginally below the original baseline accuracy of 92.40%. These substantial optimizations highlight the efficacy of FMIP in enhancing resource efficiency, thereby rendering ResNet50 more suitable for deployment in environments with limited computational resources (see Table 2).

TABLE II. TABLE 2 RESNET50 PRUNING PERFORMANCE ON IMAGENET

	ResNet50 on ImageNet datasets (Prune results)			
Approach	Baseline (%)	FLOPs Saved (%)	Pruned (%)	Accuracy (%)
[20]	-	68.95	72.94	91.91
[22]	-	45.96	-	90.71
FMIP	92.40	71.55	83.11	92.03
[21]	-	56.96	83.14	90.94

6 Conclusion

In conclusion, our proposed study exhibits the effectiveness of Feature Map Importance Pruning (FMIP) as a method for optimizing (CNNs). By systematically analyzing and eliminating redundant feature maps according to their contributions to the model's output, FMIP considerably shrinks computational costs without conceding performance. Our experiments on LeNet, VGG16, and ResNet50 networks utilizing MNIST, CIFAR10, and ImageNet datasets (see Figure 2(a,b) reveal that FMIP can effectively streamline model architecture while preserving vital features. Explicitly, we perceived a reduction in computational cost with only an insignificant reduction in accuracy on average across the evaluated models.

These results emphasize the capability of FMIP to make CNNs further practical for utilization in resource-constrained environments, such as mobile devices and embedded systems. Future research could examine the application of FMIP to other CNN architectures and datasets, as well as investigate potential combinations with other optimization methods to further boost efficiency and performance.

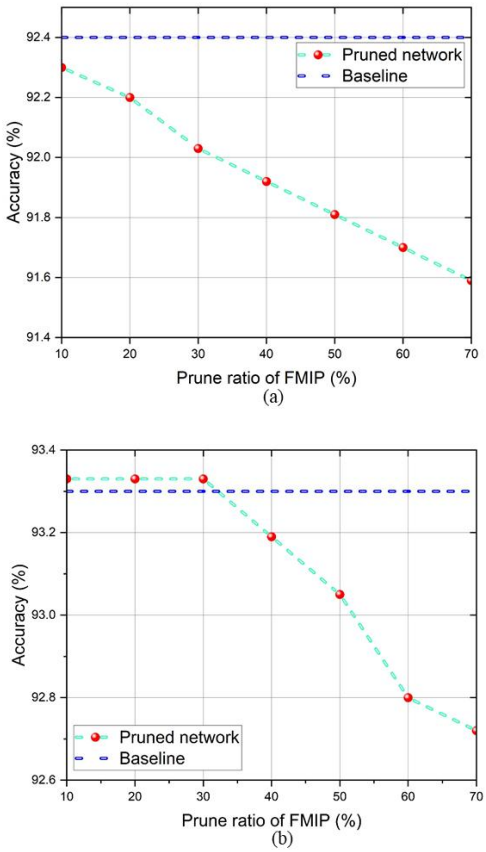

Figure 2 Pruned ResNet50 on ImageNet (a) and VGG16 on CIFAR10 (b) datasets with variable prune ratio

7 Acknowledgment

I would like to express my heartfelt thanks to my professors and friends for their invaluable support and insights during the preparation of this paper. I also appreciate the resources and fa cilities provided by the University of Science and Technology of China, which greatly contributed to this research

8 References

[1] L. Zhang, Z. Sheng, Y. Li, Q. Sun, Y. Zhao, and D. Feng, “Image object detection and semantic segmentation based on convolutional neural network,” Neural Comput. Appl., vol. 32, no. 7, pp. 1949–1958, 2020, doi: 10.1007/s00521-019-04491-4.

[2] R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation,” Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit., pp. 580–587, 2014, doi: 10.1109/CVPR.2014.81.

[3] A. Kumar, S. Wang, A. M. Shaikh, H. Bilal, B. Lu, and S. Song, “Building on prior lightweight CNN model combined with LSTM-AM framework to guide fault detection in fixed-wing UAVs,” Int. J. Mach. Learn. Cybern., vol. 15, no. 9, pp. 4175–4191, 2024, doi: 10.1007/s13042-024-02141-3.

[4] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit., vol. 2016–Decem, pp. 770–778, 2016, doi: 10.1109/CVPR.2016.90.

[5] W. Fang, F. Zhang, V. S. Sheng, and Y. Ding, “A method for improving CNN-based image recognition using DCGAN,” Comput. Mater. Contin., vol. 57, no. 1, pp. 167–178, 2018, doi: 10.32604/cmc.2018.02356.

[6] A. Kumar, B. Yin, A. K. Bhatia, A. K. Bhatia, and A. Rohra, “Structure Level Pruning of Efficient Convolutional Neural Networks with Sparse Group LASSO,” vol. 12, no. 5, 2022, doi: 10.18178/ijmlc.2022.12.5.1111.

[7] A. M. Shaikh, Y. B. Zhao, A. Kumar, M. Ali, and Y. Kang, “Efficient Bayesian CNN Model Compression using Bayes by Backprop and L1-Norm Regularization,” Neural Process. Lett., vol. 56, no. 2, pp. 1–19, 2024, doi: 10.1007/s11063-024-11593-1.

[8] A. Kumar, B. Yin, A. M. Shaikh, M. Ali, and W. Wei, “CorrNet: pearson correlation based pruning for efficient convolutional neural networks,” Int. J. Mach. Learn. Cybern., vol. 13, no. 12, pp. 3773–3783, 2022, doi: 10.1007/s13042-022-01624-5.

[9] S. Han, H. Mao, and W. J. Dally, “Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding,” pp. 1–14, 2015, doi: abs/1510.00149/1510.00149.

[10] A. Kumar, A. M. Shaikh, Y. Li, H. Bilal, and B. Yin, “Pruning filters with L1-norm and capped L1-norm for CNN compression,” Appl. Intell., 2021, doi: 10.1007/s10489-020-01894-y.

[11] Y. He, X. Zhang, and J. Sun, “Channel Pruning for Accelerating Very Deep Neural Networks,” Proc. IEEE Int. Conf. Comput. Vis., vol. 2017–Octob, pp. 1398–1406, 2017, doi: 10.1109/ICCV.2017.155.

[12] H. Hu, R. Peng, Y.-W. Tai, and C.-K. Tang, “Network Trimming: A Data-Driven Neuron Pruning Approach towards Efficient Deep Architectures,” 2016, [Online]. Available: http://arxiv.org/abs/1607.03250

[13] M. Jaderberg and T. Green, “Population Based Training of Neural Networks”.

[14] R. Abbasi-asl and B. Yu, “Structural Compression of Convolutional Neural Networks”.

[15] C. Fernando et al., “PathNet : Evolution Channels Gradient Descent in Super Neural Networks”.

[16] M. Lin et al., “HRank: Filter Pruning using High-Rank Feature Map”, Accessed: Oct. 05, 2024. [Online]. Available: https://github.com/lmbxmu/HRank.

[17] H. Zhuo, X. Qian, Y. Fu, H. Yang, and X. Xue, “SCSP: Spectral Clustering Filter Pruning with Soft Self-adaption Manners,” Jun. 2018, Accessed: Oct. 05, 2024. [Online]. Available: https://arxiv.org/abs/1806.05320v1

[18] A. Kumar, B. Yin, A. M. Shaikh, M. Ali, and W. Wei, "CorrNet: pearson correlation based pruning for efficient convolutional neural networks," Int. J. Mach. Learn. Cybern., vol. 13, no. 12, pp. 3773–3783, Dec. 2022, doi: 10.1007/S13042-022-01624-5/FIGURES/7.

[19] C. Zhao, B. Ni, J. Zhang, Q. Zhao, W. Zhang, and Q. Tian, "Variational convolutional neural network pruning," in Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2019. doi: 10.1109/CVPR.2019.00289.

[20] Z. Huang and N. Wang, "Data-Driven Sparse Structure Selection for Deep Neural Networks," in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 2018. doi: 10.1007/978-3-030-01270-0_19.

[21] S. Lin et al., "Towards optimal structured CNN pruning via generative adversarial learning," in Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2019. doi: 10.1109/CVPR.2019.00290.

[22] S. Lin, R. Ji, Y. Li, Y. Wu, F. Huang, and B. Zhang, "Accelerating convolutional networks via global & dynamic filter pruning," in IJCAI International Joint Conference on Artificial Intelligence, 2018. doi: 10.24963/ijcai.2018/336.