IFP: Filter Pruning via Information Flow for Deep Neural Networks Acceleration

Chi Xu [1]

Abstract

To facilitate the deployment of convolutional neural networks on resource-limited devices, filter pruning has emerged as an effective strategy because of its enabled practical acceleration. Evaluating the importance of filters is a crucial challenge in this field. Most existing works on filter pruning assess the importance of filters using pairwise measures such as Euclidean distance and cosine correlation, which may not capture the global information within the layer. In this paper, we propose a novel filter pruning method, IFP, which utilizes a graph-based approach to simulate the information flow among filters, thereby capturing the total information within each layer and assessing how much information is allocated to each filter. Assuming mutual information among filters, we model each filter as a linear combination of its peers and construct a directed graph accordingly. Then, a random walk is conducted on the graph, and the information distribution among filters is measured by the stable distribution upon convergence. Filters with less information distribution are interpreted as containing less useful information and can be pruned with minimal impact. We conduct image classification on CIFAR-10 and ImageNet to demonstrate the superiority of our IFP over the state-of-the-arts. For example, on CIFAR-10, IFP removes 80.28% FLOPs and 82.5% parameters from ResNet-110 with even 0.20% accuracy improvements. On ImageNet, it removes 70.2% FLOPs and 64.8% parameters from ResNet-50 with only 1.7% top-5 accuracy drops.

1. Introduction

Deep Convolutional Neural Networks (DCNNs) have revolutionized the field of computer vision by achieving state-of-the-art performance in various tasks (Simonyan & Zisserman, 2014) (He et al., 2015) (Szegedy et al., 2014). However, deploying resource-heavy CNNs on devices with limited computational and storage capacities presents significant challenges. Consequently, numerous studies have focused on model compression and CNN acceleration, mainly including network pruning (He et al., 2018b) (Lin et al., 2020) (Sui et al., 2021), model quantization (Liu et al., 2018) (Qin et al., 2020) (Liu et al., 2020), low-rank decomposition (Jaderberg et al., 2014) (Lin et al., 2016) and knowledge distillation (Shen et al., 2019) (Hinton et al., 2015). Among them, network pruning has been widely studied due to its easy implementation and effective results.

Recent developments on pruning can be divided into two categories, i.e., weight pruning and filter pruning. Weight pruning removes individual filter weights, creating a sparse network that often requires special hardware to achieve acceleration, as it cannot efficiently utilize standard BLAS libraries. In contrast, filter pruning, which compresses the model by directly removing selected filters, maintains regular structures and is widely used due to its ability to enhance acceleration on general-purpose hardware.

Generally, there are two essential issues in the filter pruning, i.e., the layer importance measurement and the filter importance measurement. For the first issue, layer importance measurement is related to the per-layer pruning rate.Existing works such as (Chin et al., 2018) (Lin et al., 2022) (Suau et al., 2018) utilize different weight-oriented strategies to evaluate the importance of each convolutional layer based on pre-trained models. For the second issue, the filter importance measurement identifies which filters in the pre-trained model should be preserved and inherited to initialize the pruned network structure. Previous works (Ye et al., 2018) (He et al., 2018a) performs filter pruning by following the "smaller-norm-less-important" criterion, which believes that filters with smaller norms can be pruned safely due to their less importance. However this criterion seems too simple and is not always true. To address this, (He et al., 2019) assumes the filters that are close to the geometric median are redundant, which is implemented by calculating the distance between filters pairs. (Joo et al., 2021) proposes Linearly Replaceable Filter (LRF), which suggests that a filter that can be ap-

[1]Data Science and Information Technology Research Center, Tsinghua-Berkeley Shenzhen Institute, Tsinghua University, China.

proximated by the linear combination of other filters is replaceable. (Lin et al., 2022) introduces a recommendation-based filter selection scheme where each filter recommends a group of its closest filters. More recently, graph-driven methods have been developed to identify important filters and achieve competitive performance. For example, Li et al. (Li et al., 2023) utilize a graphical model to represent the similarity relationships between the output feature maps of filters, while Shi et al. (Shi et al., 2023) introduce von Neumann graph entropy as a novel measure for filter importance. However, most of these approaches predominantly rely on similarity metrics like Euclidean distance or cosine similarity to measure the similarity between filters, which only considers the pairwise relationships between filters, overlooking the interactions and information sharing among them.

To address the limitations previously discussed, we proposed a novel method to evaluate the importance of filters by focusing on the flow of information between them. Recognizing that filters can have mutual information, our approach assumes that each filter potentially shares parts of its information with others. We model these interactions by linearly reconstructing each filter using its peers. From this reconstruction, the coefficients indicate the extent of information shared between a feature map and its peers, while the residuals represent the feature map's unique information. Proceeding with this approach, we construct a graph where each node represents a feature map, and the edges between them are weighted by the coefficients from the linear combinations. This graph representation allows us to apply network analysis techniques to further understand the dynamics of information sharing. Feature maps that act as central nodes in this graph could be considered as hubs of information distribution, playing pivotal roles in the layer. More details will be shown in Section 3.

In summary, the technical contributions include: 2

- Contribution 1. We introduce a unique framework that assesses filter importance based on information sharing within the network. This methodology goes beyond traditional pairwise comparison metrics, incorporating a holistic view of how filters interact and contribute to the network's functionality.
- Contribution 2. By translating filter relationships into a graph structure, our approach leverages network analysis techniques to unearth deeper insights into the informational architecture of the network. This enables us to identify key filters that serve as central hubs, thus elucidating their roles in information processing and distribution more clearly.
- Contribution 3. Applying our proposed IFP to different model pruning tasks on CIFAR and ImageNet datasets, extensive experiments demonstrate the effectiveness of our pruning strategy, which outperforms other similar algorithms in achieving high compression rates while maintaining high accuracy.

2. Related Work

This section explores seminal works in filter pruning, delineating between data-driven and data-independent methodologies before discussing advancements in graph-based techniques.

2.1. Data-Driven Pruning Methods

Data-driven pruning methods rely on training data to determine the pruned filters. ThiNet (Luo et al., 2017) adopts the statistics information from the next layer to guide the filter selections. (Liu et al., 2017) introduces sparsity in the scaling factors of batch normalization layers to identify and prune less significant channels during training. HRank (Lin et al., 2020) discovers that weights corresponding to feature maps with high rank contain more important information and therefore need to be retained in the pruning process. Beyond this CHIP (Sui et al., 2021) explores the importance of filters via using intra-channel information and FPEI (Wang et al., 2021) defines filter importance according to the entropy of the corresponding feature maps.

2.2. Data-Independent Pruning Methods

These approaches often leverage structural characteristics of the network or predefined rules. (Li et al., 2017) utilizes an l_1-norm criterion to prune unimportant filters. (He et al., 2018a) proposes to select filters with a l_2-norm criterion and prune those selected filters in a soft manner. (He et al., 2019) evaluates the redundancy of filters by measuring their distance from the geometric median of the filter group within each layer. (Lin et al., 2022) proposes Cross-Layer Ranking to decide the pruning rate of each layer and k-Reciprocal Nearest Filter selection to identify the most important filters.

2.3. Graph-Based Approaches

Graph-based methods represent an innovative frontier in filter pruning, utilizing graph theory to analyze the complex interactions between filters or layers. Techniques like the one proposed by Li et al. (Li et al., 2023) utilize graphical models to depict the similarity relationships between filters, whereas Shen et al. (Shen et al., 2019) introduced the use of von Neumann graph entropy to quantify filter importance based on their centrality in the graph structure.

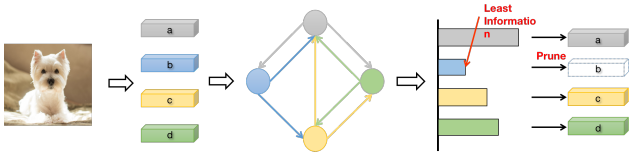

Figure 1. figure

3. Method

3.1. Linear combination of filters

To facilitate a balanced analysis of information flow across filters, each filter k_i^j within a convolutional layer is standardized by normalizing its L_2 norm. This step ensures all filters contribute equally to the network's dynamics without any bias from varying magnitudes.

The normalization process is defined mathematically as:

$$\tilde{k}_j = \frac{k_j}{\|k_j\|_2}, \tag{1}$$

where $\tilde{k}_j$ represents the normalized filter, ensuring each has a unit norm ($\|\tilde{k}_j\|_2 = 1$).

To accurately model the dynamics of information flow within the CNN, we examine the linear relationships between filters in each layer. Each filter $\tilde{k}_j$ can be approximated by a linear combination of other filters in the same layer:

$$\tilde{k}_j = \sum_{l \neq j} \lambda_{j,l} \tilde{k}_l + \varepsilon_j, \tag{2}$$

$$\min L_j = \min \left\| \tilde{k}_j - \sum_{l \neq j} \lambda_{j,l} \tilde{k}_l \right\|^2 \tag{3}$$

Then, by differentiating with respect to each $\lambda_{j,l}$ for all l, we obtain the following equation:

$$\frac{\partial L_j}{\partial \lambda_{j,l}} = 2\tilde{k}_j^T \left(\tilde{k}_j - \sum_{l \neq j} \lambda_{j,l} \tilde{k}_j \right) = 0. \tag{4}$$

Then, it becomes a system of linear equations with $n-1$ variables and $n-1$ equations, and it can be solved by matrix computation. After we obtain all the λ, we can easily calculate all the approximation errors $\varepsilon_1, \ldots, \varepsilon_n$.

3.2. Construction of directed weighted graph

Based on linear combination, we establish a directed weighted graph denoted by $\mathcal{G}$. In this graph, each node represents a filter, and the directed edges between nodes are weighted according to the absolute values of the coefficients $\lambda_{j,l}$, which reflect the strength of connection between the filters. Given that not all relationships carry equal significance, we refine our approach by focusing only on the most substantial interactions. Specifically, we select only the top 30% of the coefficients $\lambda_{j,l}$ by magnitude for each filter, as these represent the strongest ties in information sharing. This threshold ensures that we only consider interactions that have a meaningful impact on the information dynamics, dismissing weaker connections.

The weight $W_{j,l}$ of a directed edge from filter j to filter l is defined as follows:

$$W_{j,l} = \begin{cases} |\lambda_{j,l}| & \text{if } \lambda_{j,l} \text{ is among the top 30\%} \\ & \text{of coefficients for filter } j, \\ 0 & \text{otherwise.} \end{cases} \tag{5}$$

This representation allows us to capture the mutual information among filters from a information flow perspective.

3.3. Information Flow as a Randomwalk on the Graph

The weights on the graph's edges reflect the strength of connections between filters, and from the perspective of information flow, they suggest the extent to which information from one filter is disseminated to others. The underlying intuition is that information is propagated from each node to others in a manner governed by the strength of their connections. As the process unfolds, filters that emerge as high-value nodes are seen to serve as hubs, aggregating and distributing mutual information extensively. Consequently, these nodes can be considered as central in the network's architecture, playing a significant role in its functionality

To simulate the information transfer process, we propose conducting a random walk across the graph. In this model, each node (representative of a filter) has a certain probability of 'jumping' or passing information to other nodes, mirroring the real-world interactions between them. This random walk embodies the process of information exchange and distribution, allowing us to trace and quantify the flow of information and identify the most influential filters within the network. To conduct the randomwalk, we need to define the transition matrix

Transition Matrix

The transition matrix, denoted by $\mathbf{P}$, is a square matrix used to describe the transitions of a Markov chain. Its elements P_{ij} represent the probability of moving from node i to node j in one step of the walk. In our framework, we introduce a unique aspect of the transition matrix $\mathbf{P}$ to account for the intrinsic information contained within each filter. Specifically, we define the self-transition probability $P_{j,j}$ as the L_2 norm of the residual for filter j, denoted by $\|\epsilon_j\|_2$. This

self-loop represents the filter's retention of its unique information. Formally, we express this as:

$$P_{jj} = \|\epsilon_j\|_2 \tag{6}$$

With P_{jj} established, we then allocate the remaining transition probabilities to reflect the distribution of shared information. For transitions from filter j to a different filter l, the probability is proportional to the absolute weight of the edge between them, adjusted for the self-transition. This ensures that the total outgoing transition probability from any filter j, including the self-transition, sums to one. Thus, the transition probabilities for j to l ($j \neq l$) are calculated as:

$$P_{jl} = \frac{W_{jl}}{\sum_{k \neq j} W_{jk}} \cdot (1 - \|\epsilon_j\|_2) \tag{7}$$

This revised definition of the transition matrix $\mathbf{P}$ captures the dual nature of information flow: the propagation of shared information across filters and the preservation of unique information within each filter.

Random Walk

Once the transition matrix $\mathbf{P}$ has been defined, we utilize it to simulate the random walk on the graph. This process involves repeatedly applying the matrix to an initial probability distribution over the nodes, effectively modeling the step-by-step transfer of information across the network. At each step, the probability distribution vector, denoted by $\mathbf{v}$, is updated according to the equation:

$$\mathbf{v}^{(t+1)} = \mathbf{P}\mathbf{v}^{(t)}, \tag{8}$$

where $\mathbf{v}^{(t)}$ represents the distribution of the random walker's location at step t. The entries of $\mathbf{v}$ sum to one, ensuring that they represent a valid probability distribution. The process is repeated until the distribution converges to a stable state, where further applications of $\mathbf{P}$ result in no significant change in $\mathbf{v}$. This stable state is known as the stationary distribution or the steady-state distribution of the random walk, denoted by $\mathbf{v}^*$. Mathematically, the stationary distribution satisfies the equation:

$$\mathbf{v}^* = \mathbf{P}\mathbf{v}^*, \tag{9}$$

indicating that once the distribution reaches stability, it remains unchanged under the dynamics governed by $\mathbf{P}$. The stationary distribution, $\mathbf{v}^*$, offers critical insights into the network by reflecting the long-term behavior of the random walk. Filters with higher probabilities in $\mathbf{v}^*$ serve as major hubs of information flow, underscoring their pivotal roles within the network's functionality. Based on the probability values in $\mathbf{v}^*$, we can determine which filters are essential and which may be pruned to streamline the network without significantly impacting its performance.

Algorithm 1 Information Flow Pruning (IFP) for CNN Compression

Input: A pre-trained CNN with L layers and their corresponding kernels K_i; the desired number of filters to be preserved c_i.
Output: A pruned network with information-critical filters retained.
Initialize an empty list $kept\_filters$ to keep track of filters to retain in each layer.
for $i = 1$ to L **do**
    Normalize each filter k_j in layer l to have unit norm, $\tilde{k}_j = \frac{k_j}{\|k_j\|_2}$.
    for $j = 1$ to n_i **do**
        Compute the linear representation coefficients $\lambda_{j,l}$ and residuals ε_j via Equation 2.
    end for
    Construct the transition matrix P via Equation 6 and 7.
    Compute the stable distribution v^* via Equation 9.
    Select the top c_i filters based on v^* and append to $kept\_filters$.
end for
Prune the network by removing filters not in $kept\_filters$.
Optionally fine-tune the network to recover performance.
return Pruned network

4. Experiments

4.1. Experimental Settings

Baselines Models and Datasets. To demonstrate the effectiveness and generality of our proposed channel independence-based approach, we evaluate its pruning performance for various baseline models on different image classification datasets. To be specific, we conduct experiments for three CNN models (ResNet-56, ResNet-110 and VGG-16) on CIFAR-10 dataset [24]. Also, we further evaluate our approach and compare its performance with other state-of-the-art pruning methods for ResNet-50 model on large-scale ImageNet dataset [5].

Pruning and Fine-tuning Configurations. We conduct our empirical evaluations on Nvidia Tesla V100 GPUs with PyTorch 1.7 framework. To determine the importance of each filter, we randomly sample 5 batches (640 input images) to calculate the average channel independence of each feature map in all the experiments. After perform-

ing the channel independence-based filter pruning, we then perform fine-tuning on the pruned models with Stochastic Gradient Descent (SGD) as the optimizer. To be specific, we perform the fine-tuning for 300 epochs on CIFAR-10 datasets with the batch size, momentum, weight decay and initial learning rate as 128, 0.9, 0.05 and 0.01, respectively. On the ImageNet dataset, fine-tuning is performed for 180 epochs with the batch size, momentum, weight decay and initial learning rate as 256, 0.99, 0.0001 and 0.1, respectively.

5. Conclusions & Discussion

In this paper, we presented IFP, a novel filter pruning method for accelerating deep convolutional neural networks (CNNs) by leveraging information flow. Unlike traditional pairwise comparison methods, our approach evaluates filter importance based on a graph-based representation of mutual information sharing among filters within a layer. By modeling filters as a linear combination of their peers and constructing a directed weighted graph, we capture the total information within each layer and assess the centrality of each filter through a random walk process.

Our experimental results on CIFAR-10 and ImageNet datasets demonstrate the superiority of IFP over state-of-the-art methods. Specifically, IFP achieves significant reductions in FLOPs and parameters while maintaining or even improving accuracy. For instance, IFP removes 80.28% FLOPs and 82.5% parameters from ResNet-110 with a 0.20% accuracy improvement on CIFAR-10, and it removes 70.2% FLOPs and 64.8% parameters from ResNet-50 with only a 1.7% top-5 accuracy drop on ImageNet.

These results highlight the effectiveness of our proposed method in identifying and pruning less important filters, thus enabling practical acceleration of CNNs on resource-limited devices. Future work could explore the application of IFP to other types of neural network architectures and further optimize the pruning and fine-tuning processes to achieve even higher compression rates and performance gains.

6. Author Contribution & Acknowledgement

References

Chin, T.-W., Zhang, C., and Marculescu, D. Layer-compensated pruning for resource-constrained convolutional neural networks. *arXiv preprint arXiv:1810.00518*, 2018.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2015. doi: 10.1109/cvpr.2016.90.

He, Y., Kang, G., Dong, X., Fu, Y., and Yang, Y. Soft filter pruning for accelerating deep convolutional neural networks. *arXiv preprint arXiv:1808.06866*, 2018a.

He, Y., Liu, P., Wang, Z., Hu, Z., and Yang, Y. Filter pruning via geometric median for deep convolutional neural networks acceleration. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4335–4344, 2018b. doi: 10.1109/CVPR.2019.00447.

He, Y., Liu, P., Wang, Z., Hu, Z., and Yang, Y. Filter pruning via geometric median for deep convolutional neural networks acceleration. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 4340–4349, 2019.

Hinton, G., Vinyals, O., and Dean, J. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

Jaderberg, M., Vedaldi, A., and Zisserman, A. Speeding up convolutional neural networks with low rank expansions. In *British Machine Vision Conference 2014*. BMVA Press, 2014.

Joo, D., Yi, E., Baek, S., and Kim, J. Linearly replaceable filters for deep network channel pruning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pp. 8021–8029, 2021.

Li, H., Kadav, A., Durdanovic, I., Samet, H., and Graf, H. P. Pruning filters for efficient convnets. In *International Conference on Learning Representations*, 2017.

Li, J., Shao, H., Zhai, S., Jiang, Y., and Deng, X. A graphical approach for filter pruning by exploring the similarity relation between feature maps. *Pattern Recognition Letters*, 166:69–75, 2023.

Lin, M., Ji, R., Wang, Y., Zhang, Y., Zhang, B., Tian, Y., and Shao, L. Hrank: Filter pruning using high-rank feature map. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1526–1535, 2020. doi: 10.1109/cvpr42600.2020.00160.

Lin, M., Cao, L., Zhang, Y., Shao, L., Lin, C.-W., and Ji, R. Pruning networks with cross-layer ranking & k-reciprocal nearest filters. *IEEE Transactions on neural networks and learning systems*, 2022.

Lin, S., Ji, R., Guo, X., and Li, X. Towards convolutional neural networks compression via global error reconstruction. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, pp. 1753–1759. IJCAI/AAAI Press, 2016.

Table 1. Experimental results on CIFAR-10 dataset.

Method	Top-1 Accuracy (%)			Params (↓%)	FLOPs (↓%)
	Baseline	Pruned	Δ		
			VGG-16		
HRank (2020)	93.96	93.43	-0.53	2.51M (82.9%)	145.61M (53.5%)
CHIP (2021)	93.96	93.86	-0.10	2.76M (81.6%)	131.17M (58.1%)
IFP (Ours)	93.96	**94.15**	+0.19	2.76M (81.6%)	131.17M (58.1%)
HRank (2020)	93.96	91.23	-2.73	1.78M (92.0%)	73.70M (76.5%)
CHIP (2021)	93.96	93.18	-0.78	1.90M (87.3%)	66.95M (78.6%)
IFP (Ours)	93.96	**93.73**	-0.23	1.90M (87.3%)	66.95M (78.6%)
			ResNet-56		
HRank (2020)	93.26	93.17	-0.11	0.49M (42.4%)	62.72M (50.0%)
CLR-RNF (2022)	93.26	93.27	+0.01	0.38M (55.5%)	54.00M (57.3%)
IFP (Ours)	93.26	**93.78**	+0.52	0.38M (55.5%)	54.00M (57.3%)
HRank (2020)	93.26	90.72	-2.54	0.27M (68.1%)	32.52M (74.1%)
CHIP (2021)	93.26	92.05	-1.21	0.24M (71.8%)	34.79M (72.3%)
IFP (ours)	93.26	**92.62**	-0.64	0.24M (71.8%)	34.79M (72.3%)
			ResNet-110		
GAL (2019)	93.50	92.74	-0.76	0.95M (44.8%)	130.20M (48.5%)
HRank (2020)	93.50	92.65	-0.85	0.53M (68.7%)	79.30M (68.6%)
CLR-RNF (2022)	93.50	93.71	+0.21	0.53M (69.1%)	86.80M (66.0%)
IFP (Ours)	93.50	**94.39**	+0.83	0.53M (69.1%)	86.80M (66.0%)
CHIP (2021)	93.50	93.63	+0.13	0.54M (68.3%)	71.69M (71.6%)
IFP (Ours)	93.50	**93.70**	+0.20	0.32M (82.5%)	50.29M (80.28%)

Liu, Z., Li, J., Shen, Z., Huang, G., Yan, S., and Zhang, C. Learning efficient convolutional networks through network slimming. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017.

Liu, Z., Wu, B., Luo, W., Yang, X., Liu, W., and Cheng, K.-T. Bi-real net: Enhancing the performance of 1-bit cnns with improved representational capability and advanced training algorithm. In *European Conference on Computer Vision*, pp. 747–763. Springer, 2018.

Liu, Z., Shen, Z., Savvides, M., and Cheng, K.-T. Reactnet: Towards precise binary neural network with generalized activation functions. In *European Conference on Computer Vision*, pp. 143–159. Springer, 2020.

Luo, J.-H., Wu, J., and Lin, W. Thinet: A filter level pruning method for deep neural network compression. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pp. 5058–5066, Oct 2017.

Qin, H., Gong, R., Liu, X., Shen, M., Wei, Z., Yu, F., and Song, J. Forward and backward information retention for accurate binary neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2247–2256, 2020.

Shen, Z., He, Z., and Xue, X. Meal: Multi-model ensemble via adversarial learning. In *The Thirty-Third AAAI Conference on Artificial Intelligence*, pp. 4886–4893. AAAI Press, 2019.

Shi, C., Hao, Y., Li, G., and Xu, S. Vngep: Filter pruning based on von neumann graph entropy. *Neurocomputing*, 528:113–124, 2023.

Simonyan, K. and Zisserman, A. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

Suau, X., Zappella, L., Palakkode, V., and Apostoloff, N. Principal filter analysis for guided network compression. *arXiv preprint arXiv:1807.10585*, 2, 2018.

Sui, Y., Yin, M., Xie, Y., Phan, H., Aliari Zonouz, S., and Yuan, B. Chip: Channel independence-based pruning for compact neural networks. *Advances in Neural Information Processing Systems*, 34:24604–24616, 2021.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S. E., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. Going deeper with convolutions. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1–9, 2014. doi: 10.1109/CVPR.2015.7298594.

Wang, J., Jiang, T., Cui, Z., and Cao, Z. Filter pruning with a feature map entropy importance criterion for convolution neural networks compressing. *Neurocomputing*, 461:41–54, 2021. doi: https://doi.org/10.1016/j.neucom.2021.07.034.

Ye, J., Lu, X., Lin, Z., and Wang, J. Z. Rethinking the smaller-norm-less-informative assumption in channel pruning of convolution layers. *arXiv preprint arXiv:1802.00124*, 2018.