Dynamic Structured Pruning With Novel Filter Importance and Leaky Masking Based on Convolution and Batch Normalization Parameters

INCHEON CHO, EUNSEOP SHIN, MUHAMMAD SALMAN ALI, AND SUNG-HO BAE, (Member, IEEE)
Department of Computer Science and Engineering, Kyung Hee University, Yongin-si 17104, Republic of Korea
Corresponding author: Sung-Ho Bae (shbae@khu.ac.kr)
This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science, ICT & Future Planning (2018R1C1B3008159).

ABSTRACT Various pruning methods have been proposed to solve the overparameterized problem in deep neural networks. Most of the structured pruning methods have used magnitude-based filter importance to remove unnecessary filters. Usually, Convolutional Neural Networks (CNN) consist of blocks that proceed with batch normalization (BN) operations after convolutional (Conv) operations. Each element is calculated through cooperation between Conv and BN, so both Conv and BN parameters must be considered together in pruning. However, previous pruning methods independently used the norm of parameters, either Conv weights or BN scales, as filter importance, ignoring these CNN structures. With this intuition, we propose a new magnitude-based filter importance method that considers both Conv weights and BN scales, and provide evidence of importance through experimental analysis on the rank of the feature map as well as mathematical analysis on the feature distortion. Furthermore, in recent works, dynamically applying a mask to recover weights has enabled more accurate pruning. However, when the weights are multiplied by zero mask values, the gradients become zero, which prevents updating the weights. We name this problem the zero gradient transferring. To solve this problem, we propose a leaky masking method that replaces the zero value of the mask with a positive constant. As a result, we solve the zero gradient transferring through the leaky masking method, enabling more accurate dynamic structured pruning. We denote our proposed method as CoBaL that incorporates Conv and BN based filter importance and Leaky mask into dynamic structured pruning. Experimental results show that our CoBaL compresses 50.09% parameters and 32.51% FLOPs on ResNet 56 with the CIFAR-10 with slightly improved accuracy. Also, we possess comparable results on ImageNet with 50.74% and 29.38% reduction in the number of parameters and FLOPs at ResNet18, respectively.

INDEX TERMS Deep learning, model compression, lightweight, pruning, zero gradient transferring, leaky masking, CoBaL.

I. INTRODUCTION

Deep Neural Networks (DNN) based on Convolutional Neural Networks (CNN) have shown state-of-the-art performance in computer vision applications, like image classification [1]–[3], object detection [4], [5], and segmentation [6], [7]. However, with this improved performance, increasing storage capacity and computational cost require expensive hardware resources. However, CNN-based algorithms require high computational and storage capacity, which requires extensive hardware resources. These problems made it difficult to deploy DNNs in limited-resource environments, such as edge devices and mobile appliances. To remedy these problems, model compression research has been briskly advanced. Model compression is typically studied in various methods such as pruning [8], [9], quantization [10], knowledge distillation [11], weight sharing [12], and compact modeling [13].

Since pruning can produce simple, intuitive, and efficient results by eliminating relatively unnecessary weights, we conducted this study with a focus on pruning. Pruning is typically classified into unstructured pruning and structured pruning, depending upon the form of weights that are being removed. Unstructured pruning removes (converts them into...
zero) the elements inside the tensor, obtaining sparse matrices. Usually, unstructured pruning guarantees minimal degradation in performance even at high pruning ratios. However, they need specialized software [14] and hardware [15] to ensure acceleration in reality. On the other hand, structured pruning does not have this constraint because it removes the entire filter. Since structured pruning is capable of practical compression of parameters and FLOPs and does not require specialized hardware and software, this paper focuses on structured pruning.

The conventional structured pruning methods statistically apply masking to remove weights. However, the static methods are relatively inaccurate because the removed weights are never reused in pruning process. To solve this phenomenon and prune the weights more precisely, recent works have focused on updating the mask dynamically. Dynamic methods have generated learnable masks [16]–[18] or modified gradients [19], [20] to update the masks dynamically. With this flexibility, recent dynamic methods achieved higher performance than static methods. Since the dynamic method showed more accurate and stable performance, we proceed with the study focusing on the dynamic method.

In general, dynamic structured pruning has evolved in two directions, (1) exploring optimal filter importance, which guarantees high performance, and (2) effectively resuscitating removed weights.

Regarding (1), we can minimize performance deterioration with increasing pruning ratios by designing the appropriate filter importance. Among various filter importance methods, the magnitude-based method has been universally used. Most CNNs architecture consists of layers where a single layer performs batch normalization (BN) after convolutional (Conv) operations. Therefore the output of each layer is generated through the computational cooperation of Conv and BN. In this regard, we argue that the both Conv and BN parameters can be considered together for calculating importance in a more accurate manner. However, the conventional magnitude-based methods have considered only one parameter type, i.e., Conv weights or BN scales. Furthermore, either experimental or mathematical analysis on how magnitude-based methods perform well has not thoroughly been conducted.

Regarding (2), dynamically updating masks instead of using statically fixed masks ensures more accurate performance in pruning. Recent dynamic methods update the mask with learnable parameters while utilizing modified gradients. First, we identified a problem when extending dynamic pruning from Conv to BN in ReLU. Due to the characteristic of ReLU, it is difficult for the feature maps with zero values to have gradients. Furthermore, regardless of activation function types, the gradients of the weights located in low-level layers (i.e. close to the input layer) do not include whole information of high-level layers (i.e. close to output layer) which are multiplied by zero masks. That is, since the gradients of the weights are calculated by multiplication of gradients for the feature maps, the zero gradients of the feature maps are ignored in backpropagation process.

With these motivations, we analyze the existing magnitude-based filter importance methods and discover their association with the generation of informative feature maps. Based on the experimental evidence and mathematical analysis, we propose a more accurate magnitude-based filter importance method, by fusing Conv weights and BN scales. Moreover, we found that applying a binary mask often generates sparse weights and gradients through the mathematical analysis on the gradients. That is, the gradients located at zero-valued mask elements scarcely propagate during the backpropagation process. We denote these problems as the zero gradient transferring problem. Finally, to solve these problems, we propose a new leaky masking method that replaces all the zero values with positive constants in the mask. Also, the proposed leaky masking method functions as regularizing unnecessary weights and penalizing gradients. Therefore, taking advantage of higher flexibility obtained from the leaky mask, the dynamic pruning method using leaky mask can both recover removed weights and eliminate insignificant filters more precisely. In this paper, we utilize existing pruning method called DPF [19] to validate our leaky masking method. We denote our proposed method as CoBaL that incorporates Conv and BN based filter importance and Leaky mask into dynamic structured pruning.

Our comprehensive experiments demonstrate that the proposed CoBaL improved performance over existing filter pruning and model compression methods.

A. CONTRIBUTIONS

The main contributions of this work are as follows:

- From comprehensive experiments, we found out that the BN parameters can reflect the rank of the feature maps, which leads us to design new filter importance that combines magnitude of weights and BN parameters.
- Based on experimental evidence and mathematical analysis, we further analyzed the effectiveness of the proposed filter importance.
- We further propose a new leaky masking method to solve the zero gradient transferring problem by regularizing weights and transferring penalized gradients during the training process of dynamic structured pruning.
- The proposed method achieved comparable accuracy at 30% of compression ratios compared to the un-pruned baseline model with ResNets [2] (i.e., ResNet56/110) and MobileNetV1 [3] on CIFAR10 and ResNet18/50 on ImageNet without additional fine-tuning process.

II. RELATED WORK

A. STRUCTURED PRUNING

Pruning is considered an effective method to relax the over-parameterization problem and accelerate inference speed. Pruning methods can be divided into unstructured and structured methods according to the form in which the
weights are removed. First, unstructured pruning produces sparse matrices by converting the elements to zero in the tensors. In unstructured pruning, magnitude-based pruning methods [21], [22] were proposed to remove redundant individual weights. Discarding a single weight ensures high sparsity with minimum accuracy drop. However, for such a method, specialized hardware [15] and software [23] are required to achieve practical efficiency in acceleration.

Beyond conventional unstructured pruning methods, various structured pruning methods have been studied to enjoy accelerating the inference time at the cost of sacrificing a certain degree of weight sparsity. Previous structured pruning studies designed filter importance with magnitude of either the Conv weights [8], [9], [24] or the BN trainable parameters [25]–[27]. Despite most CNN architectures being configured in Conv-BN structure, they did not consider both Conv and BN simultaneously. In this paper, through experimental evidence and mathematical analysis, we show that the norm of Conv weights and BN scales play significant roles in calculating filter importance. Hence, we propose new filter importance by combining the parameters of the Conv and the BN.

### B. DYNAMIC PRUNING

Conventionally, pruning methods have been developed from the static pruning methods [21], [22] which remove the weights iteratively in a greedy manner. Static pruning methods reduce the number of parameters and FLOPs in intuitive and straightforward ways, but there is a problem that removed weights are never reused. Therefore, due to this inflexibility in pruning, they often yield mild performance degradation at the expense of compressing the model size.

To solve this problem, dynamic pruning methods have been devised, guaranteeing the recovery of the weights that were being removed. Various dynamic methods [17]–[20], [24], [28]–[31] have been proposed, and they have shown better performance than static methods. Dynamic pruning methods require masks corresponding to the weights to keep the pruning status of the weights (i.e., 0 and 1 indicate pruned and unpruned status, respectively).

Recently, [19] proposed a dynamic method to enable the update of the weights multiplied by the mask through gradient manipulation, which replaces the existing weight gradients with the gradients that occur from the weight multiplied by the mask. In other words, the gradient manipulation is to renew the weight by replacing the zero gradients with non-zero gradients. Similar to [19], [32] restored the zero weights by repeatedly conducting asymptotic pruning and fine-tuning during training. Furthermore, [31] proposed a drop and grow strategy with the magnitude of the parameters and the gradients. Meanwhile, [17], [18], [20], [29], [30] solved non-differentiable problem for the binary masks using Gumbel-Softmax [33], alternating direction method of multipliers (ADMM) [34], straight-through estimator (STE) [35], and so on. With these differentiable masking schemes, previous studies proposed an appropriate optimization method by adding a sparse term to the objective function.

Although the aforementioned dynamic methods show good performance taking advantage of recovering removed weights, we found that they still suffer from the zero gradient transferring problem. The zero gradient transferring problem is that the gradient is insufficiently transferred due to zero-valued elements in the mask. That is, zero-values in masks are multiplied by the gradients of weights/feature maps during the backpropagation process, thus zeroing the gradients. This problem inhibits the renewal of weights by the chain rule. To solve this problem, [36] proposed a random walk process which allows pruned filters to jump to be activated to restore learning ability. Similar to [36], [37] added an incremental regularization term for pruned weights to have a chance to be reactivated. Although the random walk and incremental regularization are able to recover learning ability, they may hinder converging to optimal minima. To solve these problems, we propose a leaky masking method that replaces the minimum value of the mask with a positive constant value.

### III. PROPOSED METHODS

#### A. NOTATIONS

Assume a CNN model $\tilde{y} = f(x; \Theta)$ has a set of $L$ Conv layers where $\Theta = \{w^1, w^2, \ldots, w^L\}$. $x$ and $y$ are the input and true label as $(x, y) \in D$ where $D$ is data distribution, and $\tilde{y}$ is the predicted label of $x$. The $l$-th layer’s input, output and weight are denoted by $x_{in}^l, x_{out}^l$, and $w^l \in \mathbb{R}^{c_{out} \times c_{in} \times k \times k}$, respectively. Here, $c_{out}$ is the number of output channels, $c_{in}$ is the number of input channels and $k$ is the kernel size.

#### B. RANK ANALYSIS

Delivering features to all layers without losing representational information is essential for performance preservation. To transfer informative feature maps through all layers, recent studies mitigated representational bottleneck to maximize the rank of the feature maps [38]. Based on these rank characteristics, recent work has used the rank as the criterion for pruning [39] to minimize performance degradation. We also aimed to use the rank as filter importance by exploiting these advantages. The previous methods require a forward process that feeds mini-batches of training data into the model to create feature maps to obtain the rank of the feature maps. Also, it takes an excessive amount of time to calculate rank based on the generated feature maps because calculating rank requires a large amount of computation.

To accelerate this process, we aim to find the parameters that can reflect the rank of the feature maps. First, we investigated the relevance between the BN parameters and the rank of the feature maps. We conducted 10,000 experiments depending on the BN parameter values and took the average for the normalized rank of the output feature maps. The normalized rank is the rank normalized by the number of elements in the output feature map. As shown in Figure 1, the larger the BN parameters are, the greater the normalized rank gets. This experiment motivated us to incorporate BN parameters into the feature importance metric.
we propose new filter importance considering the sign of BN and variability of elements in a layer as

\[ F_i = ||y_i||||w_i|| + a \cdot \text{sign}(\beta_i)||\beta_i|| \]  

(4)

where \( a \) is the scale value to reflect the confidence of the BN bias. Since \( ||x_{in}|| \) has variability, we normalize Eq. (3) by \( ||x_{in}|| \). It is also worth noting that we consider the sign of the BN bias in Eq. (4) as the rank of the feature map decrease when the sign of the BN bias gets negative as shown in Figure 1-(b). Nevertheless, a positive \( a \) has drawbacks, such as incurring hyper parameter search costs and increasing computational complexity of importance due to increased degrees of freedom. In this paper, we confirm stable performance when \( a = 0 \) (detailed experimental results are shown in Table 3), as well as propose the following formula for simplicity and increased utilization (no hyper parameters) of importance. Therefore, the filter importance is shown in the following equation by setting \( a \) to zero as

\[ F_i^l = ||y_i^l||w_i^l|| \]  

(5)

where \( F_i^l \) is the importance of \( i \)-th filter in \( l \)-th layer s.t. \( i \in \{1, 2, \ldots, c_{out}\} \).

D. GROUP IMPORTANCE

In structured pruning, we ensure the efficiency of memory and acceleration by eliminating filters multiplied by zero. Suppose we remove the filters without considering the residual connection which applies element-wise addition on the feature maps. In that case, it may result in the misalignment of feature maps in the residual connection. To solve the misalignment problem, the previous studies [25], [43] suggested the group importance considering the filters belonging to the residual connection by grouping them. They proposed group importance by adding all the importance of filters belonging to the same residual connection [25] and by calculating data-dependent distortion [43]. However, [25] does not reach optimal memory efficiency due to the bottleneck structure in the residual blocks, and [43] takes a lot of computational time due to calculating data-dependent distortion.

To remedy these disadvantages, we aim to prevent the bottleneck problem and calculate group importance in a fast manner. We define group importance as the mean importance in a residual connection group. Then, the group importance element \( F_i^g \) of a group \( g \) can be calculated by

\[ F_i^g = \frac{1}{n(L_g)} \sum_{l \in L_g} F_i^l \]  

(6)

where \( L_g \) is the index set of layers in \( g \) and \( n(L_g) \) indicates the number of layers in \( g \). After calculating the importance of all filters, we remove filters that are less than a threshold.

E. LEAKY MASKING METHOD

Based on the proposed filter importance, we apply the dynamic pruning approach modifying the previous method called DPF [19] which was initially designed for unstructured...
pruning. We extend DPF for dynamic structured pruning by applying a mask on BN instead of Conv to simplify structured pruning. Let \( \text{Loss} : \mathbb{R}^c \rightarrow \mathbb{R} \) be an objective function where \( c \) is the number of classes and \( m \in \{0, 1\}^d \) be a mask to the BN parameters where \( d \) is the dimension of the mask. Then, the update scheme for BN scale \( \gamma \) can be defined as

\[
\gamma_{t+1} := \gamma_t - \alpha_t \frac{\partial \text{Loss}(\mathbf{x}, \mathbf{y}; \theta_t; m_t)}{\partial \gamma_t}
\]

where \( t \) is time, \( \alpha \) is the learning rate, \( \theta \) us the model parameters, and \( \tilde{\mathbf{y}} = m \odot \gamma \) where \( \odot \) is Hadamard product. Based on the STE, replacing previous gradient part with partial derivative respecting to \( \tilde{\mathbf{y}} \) could solve gradient flow problem. However, under ReLU activation, directly applying a binary mask to BN parameters still causes the zero gradient transferring problem, thus restraining the model from being converged. As considering the characteristic of ReLU, the single layer’s partial derivative of \( x_{out} \) with respect to \( i \)-th element of \( \tilde{\mathbf{y}} \) can be calculated as

\[
\frac{\partial x_{out}}{\partial y_i} = \begin{cases} 
  z_i & m_i \neq 0 \\
  0 & m_i = 0
\end{cases}
\]

where \( x_{out} > 0 \). Based on the observation of Eq. (8) and the proxy analysis [48], the Eq. (7) can be expressed as

\[
y_{t+1} := \text{prox} \left( y_t, \alpha_t \frac{\partial \text{Loss}(\mathbf{x}, \mathbf{y}; \theta_t; m_t)}{\partial \gamma_t} \right)
\]

where \( \text{prox}(x, y) = x - \max\{|y|, 0\} \cdot \text{sign}(y) \). In the previous method [48], the information flood was determined according to the weight decay value. Compared to [48], in our method, the information flood of the gradient is determined according to the mask value. With this analysis, we observed that the zero gradient transferring problem still remained in ReLU activation. Furthermore, we also observed aforementioned problem regardless of activation function. The partial derivative of \( l+1 \)-th layer’s output with respect to \( l \)-th layer’s output can be calculated as

\[
\frac{\partial x_{out}^{l+1}}{\partial x_{out}^l} = m \odot \gamma \odot \frac{\partial S(x_{out}^l)}{\partial x_{out}^l}
\]

In Eq. (10), the gradients of low-level layers do not include information about part of \( x_{out}^{l+1} \) when the elements of \( m \) equals to zero. This problem generally occurs in pruning applying masks.

From Eq. (9) and Eq. (10), it can be seen that the gradient becomes zero when \( m_i = 0 \) that hinders optimizing the weights multiplied by zero value of the mask. To prevent this zero gradient transferring problem, we define a new leaky masking method which is denoted by \( \tilde{m} \in \{c, 1\}^d \) for some positive constant \( c \), s.t. \( 0 < c < 1 \). From Eq. (10), we can easily find the fact that the minimum value of \( \tilde{m} \) affects to gradient as a penalty.

It is noted that if the minimum value of \( \tilde{m} \) is set to be close to one, the pruning status for the weights can rapidly change during training. As a result, our dynamic pruning performance becomes similar to one-shot pruning resulting in significant performance deterioration. Conversely, a minimal value of \( \tilde{m} \) can interfere with convergence at good local optima due to the strong regularization effect in gradient. To mitigate this phenomenon, we set the minimum value of \( \tilde{m} \) to gradually decrease for stable learning. We will describe the details of this in Section IV-D3.

It is worth noting that the gradual pruning [49] is performed up to first \( E_1 - 1 \) epochs while the second learning rate decay occurs from \( E_1 \) epochs in training. We calculate the filter importance with Eq. (5) and sort the filters according to the importance. After then, the lowest importance filters within pruning ratio are multiplied with minimum value of the mask while rest of them are preserved. At the end of the gradual pruning process, we replace the \( \tilde{m} \in \{c, 1\}^d \) with \( m \in \{0, 1\}^d \) for the rest of the training process. Algorithm 1 describes the detail of the training procedure of CoBaL.

**IV. EXPERIMENTS**

In this part, we evaluate the performance of the proposed method. We validate the effectiveness of classification tasks. Experiments were held on CIFAR-10 [50] and ImageNet [51] datasets. We also conducted an ablation study to compare the effectiveness of each component.

**A. EXPERIMENTAL DETAILS**

On CIFAR-10, we chose ResNet56/110 [2] and MobileNetV1 [3] for experiments. The optimal learning rate for ResNet56/110 is 0.2; the corresponding weight decay is 1e-4. We train whole architectures using stochastic gradient descent (SGD) with Nesterov momentum [52] with the momentum of 0.9 and mini-batch size of 128 using a single GPU. We trained whole architectures for 300 epochs and decayed learning rate to be 1/10 in scale at 150, 225 epochs in training, i.e., \( E_1 \) on CIFAR-10 is 225.
TABLE 1. Comparison of the proposed method with other methods on CIFAR-10. The compared methods consist of structured pruning and compact model methods. $\Delta$Acc is the amount of change in Top-1 Accuracy. $R_{\text{params}}$ and $R_{\text{FLOPs}}$ mean how many parameters and FLOPs remained in percentage after pruning, respectively. In the notation of CoBaL-N, N indicates pruning ratio (e.g., CoBaL-30 indicates that the target pruning ratio is 30%).

| Baseline | Method | Top-1 | $\Delta$Acc | $R_{\text{params}}$ (%) | $R_{\text{FLOPs}}$ (%) |
|----------|--------|-------|-------------|------------------------|------------------------|
| ResNet56 | Baseline | 94.12 | - | 0.86M | 125.49M |
|          | GBN [25] | 93.31 | 94.07 | -0.03 | - | 66.7 | - | 43.3 |
|          | HRank [39] | 93.26 | 93.17 | -0.09 | 0.49M | 58.6 | 62.72M | 50.00 |
|          | EAGLEEye [44] | 93.26 | 94.66 | 1.5 | - | - | 62.23M | 49.59 |
|          | STRUCT-A [45] | 93.03 | 92.63 | -0.38 | 0.4M | - | 63.04M | 51.03 |
|          | DHP [46] | 92.95 | 93.58 | 0.63 | - | 50.96 | - | 58.42 |
|          | STRUCT-B [45] | 93.03 | 91.78 | -1.25 | 0.21M | - | 48.49M | 38.64 |
|          | FPGM [9] | 93.59 | 93.49 | 0.1 | - | - | 59.4M | 52.67 |
|          | HINGE [47] | 92.95 | 92.65 | -0.30 | 0.18M | 20.80 | - | 30.12M | 24.00 |
|          | COBAL-30(OURS) | 94.12 | 94.14 | 0.02 | 0.54M | 49.91 | 84.69M | 67.49 |
|          | COBAL-50(OURS) | 94.12 | 93.62 | -0.4 | 0.34M | 39.53 | 58.20M | 46.38 |

| Baseline | Method | Top-1 | $\Delta$Acc | $R_{\text{params}}$ (%) | $R_{\text{FLOPs}}$ (%) |
|----------|--------|-------|-------------|------------------------|------------------------|
| ResNet110 | Baseline | 94.31 | - | 1.73M | 255.28M |
|           | HRANK [39] | 93.50 | 94.23 | 0.73 | 1.04M | 60.6 | 148.70M | 59.8 |
|           | DHP [46] | 94.69 | 94.63 | -0.06 | - | 63.66 | - | 63.2 |
|           | COBAL-30(OURS) | 94.31 | 94.49 | 0.18 | 1.19M | 68.79 | 167.85M | 65.75 |

| Baseline | Method | Top-1 | $\Delta$Acc | $R_{\text{params}}$ (%) | $R_{\text{FLOPs}}$ (%) |
|----------|--------|-------|-------------|------------------------|------------------------|
| MobileNetV1 | Baseline | 90.88 | 90.88 | -0.1 | 1.82M | 56.52 | 27.13M | 57.50 |
| MobileNetV1 x0.75 | EAGLEEye [44] | 90.88 | 91.89 | - | 1.4M | 43.48 | 22.95M | 48.64 |
| MobileNetV1 x0.5 | COBAL-30(OURS) | 90.88 | 89.99 | -0.89 | 0.82M | 25.47 | 12.56M | 26.66 |
| MobileNetV1 x0.25 | EAGLEEye [44] | 90.88 | 91.44 | - | - | - | 12.1M | 25.65 |
| MobileNetV1 x0.25 | COBAL-93(OURS) | 90.88 | 87.32 | -3.56 | 0.22M | 6.83 | 3.34M | 7.50 |

B. CIFAR-10

Table 1 shows the comparison of different methods for ResNet56/110 and MobileNetV1 on CIFAR-10. To validate our method, we compared previous pruning methods [9], [25], [39], [44], [46], [47] and a compact model method [45]. Considering the trade-off between compression efficiency and prediction accuracy, we compare the accuracy at the similar compression efficiency level. We denote ‘CoBaL-N’ where N indicates the percent of pruning ratio. To evaluate the number of parameters, we count the parameters in all layers. For FLOPs, the total number of Multiply-Accumulations (MACs) operations in one inference is used. As shown in Table 1, under similar efficiency condition, CoBaL shows reasonable performance when compared to other state-of-the-art methods in Top-1 accuracy. Furthermore, CoBaL-30 obtains higher Top-1 accuracy compared to baseline.

C. ImageNet

We compared our method with the previous pruning methods [9], [16], [25], [39], [43], [44], [47], [55] and compact model methods [45], [56]. Table 2 shows the result on ResNet18/50 compression, where ‘F.T.’ indicates the usage of fine-tuning. We performed experiments with 30% of pruning ratios. As shown in Table 2, for ResNet18 and ResNet50, CoBaL showed comparable performance to other methods even with 50.74% and 46.33% reduction in the number of parameters, respectively. Also, CoBaL achieves the 29.38% 44.26% reduction in FLOPs for ResNet18 and ResNet50, at a similar performance level, respectively. HRank [39] is a pruning method based on rank-based filter importance.
TABLE 3. Effectiveness of the weight magnitude and BN scale in filter importance. F.I. column refers to filter importance method and Eq. column indicates equations according to F.I.

| F.I.  | Eq.      | Top-1 | Params | FLOPs  |
|-------|----------|-------|--------|--------|
| CONV  | $\|w\|$  | 92.95 | 0.24M  | 67.44M |
| BN    | $\|\gamma\|$ | 93.52 | 0.35M  | 56.24M |
| PROPOSED | Eq. (4) ($a = 0$) | 93.62 | 0.34M  | 58.20M |
| PROPOSED | Eq. (4) ($a = 1$) | 85.95 | 0.17M  | 39.28M |

Compared to HRank, CoBaL shows comparable performance in Top-1 accuracy while reducing more parameters and FLOPs.

D. ABALATION STUDY

To check the effectiveness of each proposed component, i.e., filter importance and leaky masking, we perform an ablation study. We perform all of the experiments under the same condition. However, since our method is a pruning method based on dynamic filter pruning with global importance, the resulting number of parameters and FLOPs can be different at the same target pruning ratio. We perform experiments with ResNet56 on CIFAR-10 where the pruning ratio is set to 50%. This experimental setup is ideally used in this section.

1) FILTER IMPORTANCE

We divide our importance metric into individual components (i.e the magnitude of Conv weight and BN scale) and perform experiments for individual components.

As shown in Table 3, the proposed filter importance yields the highest accuracy compared to the models trained with individual filter importance. In Table 3, we also found that the proposed filter importance with $a = 1$ in Eq. (4) shows lower performance than the proposed importance with $a = 0$ in Eq. (4)), implying that BN bias has a negative effect in measuring the importance of a filter. Therefore, we conclude that both the magnitude of Conv weight and BN scale help measuring filter importance in a complementary manner.

2) GROUP IMPORTANCE

To validate the effectiveness of the group importance, we compare the number of parameters, FLOPs, and latency with or without the group importance. In our experiments, we evaluate actual latency on a single CPU (Intel Xeon Silver 4214) and GPU (NVIDIA GeForce GTX 1080Ti). To measure the latency, we use one data sample for the forward process of a model. In Table 4, we observed that utilizing group importance ensures better performance, efficiency in the number of parameters. Although theoretical FLOPs with group importance has slightly larger than that of without the group importance, using our group importance yields fast latency in real-world. It is noticed that, without the group importance, the actual pruning process entails heavy human intervention for eliminating the pruned filters and rearranging the filter indices. On the other hand, incorporating the group importance performs the pruning automatically without any human intervention aforementioned. In this regard, we evaluate the latency under the condition that no human intervention is performed.

3) LEAKY MASKING

Table 5 shows the performance sensitivity of $c$ in the leaky mask. As shown in Table 5, adopting a positive $c$ value increases the accuracy, indicating that the proposed leaky masking method remedies the zero gradient transferring problem in the conventional masking methods.

We also empirically prove that a proposed method that gradually decreases $c$ along with $-\lambda \cdot i/E_1$ guarantees better performance, where $\lambda$ is set to 9, and $i$ is the epoch index in training. Further analysis of leaky masking method is in Appendix section.

4) PRUNING RATIO

In Figure 2, we measure Top-1 test accuracy and remaining parameters/FLOPs ratios according to the pruning ratio. As shown in Figure 2, we found that there is no accuracy drop up to 30% of the pruning ratio while considerably reducing the number of parameters and FLOPs. Reaching out to the pruning ratio of 60%, the number of parameters and
FLOPs are significantly reduced at the expense of a slight decrease in accuracy. Therefore, as the effective pruning ratio is up to 60%, we perform experiments with the pruning ratio of either 30% or 50%.

V. CONCLUSION

We explore new filter importance that considers the magnitude of both Conv weights and BN scale for structured pruning. Furthermore, our proposed leaky masking mitigates the zero gradient transferring problem by which the effectiveness of the dynamic pruning enhances. Our comprehensive experiments reveal that CoBaL outperforms the existing pruning and compact model-based methods. Our future work aims to improve our filter importance by effectively combining the BN bias into the importance metric.

APPENDIX

A. DERIVATION OF INEQUALITY

In this section, we prove the Eq. (3). With Cauchy-Schwarz inequality, Eq. (3) can be derived as

$$\|y_i S(w_i \otimes x_{in}) + \beta_i\| \leq \|y_i\| \|S(w_i \otimes x_{in})\| + \|\beta_i\|$$

Previous study [57] shows that most of the value of batch statistics usually higher than 1. With this observation, Eq. (11) can be expressed as

$$\|y_i\| \|S(w_i \otimes x_{in})\| + \|\beta_i\| \leq \|y_i\| \|w_i \otimes x_{in}\| + \|\beta_i\|$$

With Young’s convolution inequality, Eq. (12) can be derived as

$$\|y_i\| \|w_i \otimes x_{in}\| + \|\beta_i\| \leq \|y_i\| \|w_i \otimes x_{in}\| + \|\beta_i\|$$

B. APPLYING LEAKY MASKING FOR DIFFERENTIABLE MASK

To validate general effectiveness of the leaky masking method, we conducted experiments on differentiable masks. We employed Gumbel-Softmax [33] trick, which performs a differentiable approximation to a categorical random variable. On CIFAR-10, we trained ResNet56 for 200 epochs. We also utilized group pruning with differentiable masks to practically exclude zero weights after training process. In Table 6, we observed that applying the leaky masking method ensures higher performance in terms of tradeoff between efficiency and accuracy.

| MASK VALUE | TOP-1 (%) | PARAMS (M) | FLOPs (M) |
|------------|-----------|------------|-----------|
| {0, 1}     | 92.62     | 0.35M      | 75.66M    |
| {10^{-2}, 1} | 92.74     | 0.33M      | 69.20M    |
| {10^{-4}, 1} | 93.03     | 0.46M      | 61.87M    |

ACKNOWLEDGMENT

This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science, ICT & Future Planning (2018R1C1B3008159).

REFERENCES

[1] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, “Going deeper with convolutions,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2015, pp. 1–9.
[2] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2016, pp. 770–778.
[3] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, “MobileNets: Efficient convolutional neural networks for mobile vision applications,” 2017, arXiv:1704.04861.
[4] R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., Jun. 2014, pp. 580–587.
[5] S. Ren, K. He, R. B. Girshick, and J. Sun, “Faster R-CNN: Towards real-time object detection with region proposal networks,” in Proc. NIPS, 2015, pp. 91–99.
[6] J. Long, E. Shelhamer, and T. Darrell, “Fully convolutional networks for semantic segmentation,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2015, pp. 3431–3440.
[7] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, “Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 40, no. 4, pp. 834–848, Apr. 2017.
[8] H. Li, A. Kadav, I. Durdanovic, H. Samet, and H. P. Graf, “Pruning filters for efficient ConvNets,” 2016, arXiv:1608.08710.
[9] Y. He, P. Liu, Z. Wang, Z. Hu, and Y. Yang, “Filter pruning via geometric median for deep convolutional neural networks acceleration,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2019, pp. 4340–4349.
[10] S. K. Esser, J. L. McKinstry, D. Bblanli, R. Appaswamy, and D. S. Modha, “Learned step size quantization,” 2019, arXiv:1902.08153.
[11] G. Hinton, O. Vinyals, and J. Dean, “Distilling the knowledge in a neural network,” 2015, arXiv:1503.02531.
[12] Z. Wang, X. Cheng, G. Sapiro, and Q. Qiu, “ACDC: Weight sharing in atom-coefficient decomposed convolution,” 2020, arXiv:2009.02386.
[13] V. Lebedev, Y. Ganin, M. Rakubha, I. Osleedets, and V. Lempitsky, “Speeding-up convolutional neural networks using fine-tuned CP-decomposition,” in Proc. 3rd Int. Conf. Learn. Represent., (ICLR) Conf. Track, 2015, pp. 2–12.
[14] J. Park, S. Li, W. Wen, P. T. P. Tang, H. Li, Y. Chen, and A. Dubey, “Faster CNNs with direct sparse convolutions and guided pruning,” 2016, arXiv:1608.01409.
[15] S. Han, X. Liu, H. Mao, J. Pu, A. Pedram, M. A. Horowitz, and A. J. Dally, “EIE: Efficient inference engine on compressed deep neural network,” in Proc. Int. Symp. Comput. Archit., vol. 44, no. 3, 2016, pp. 243–254.
[16] S. Lin, R. Ji, Y. Li, Y. Wu, F. Huang, and B. Zhang, “Accelerating convolutional networks via global & dynamic filter pruning,” in Proc. 27th Int. Joint Conf. Artif. Intell., Jul. 2018, pp. 2425–2432.
[17] M. Kang and B. Han, “Operation-aware soft channel pruning using differentiable masks,” in Proc. Int. Conf. Mach. Learn., 2020, pp. 5122–5131.
[18] A. Kusupati, V. Ramanujan, R. Somani, M. Wortsman, P. Jain, S. Kakade, and A. Farhadi, “Soft threshold weight reparameterization for learnable sparsity,” in Proc. Int. Conf. Mach. Learn., 2020, pp. 5544–5555.
[19] T. Lin, S. U. Stich, L. Barba, D. Dmitriev, and M. Jaggi, “Dynamic model pruning with feedback,” in Proc. Int. Conf. Learn. Represent., 2019, pp. 1–22.
[20] J. Liu, Z. Xu, R. Shi, R. C. C. Cheung, and H. K. H. So, “Dynamic sparse training: Find efficient sparse network from scratch with trainable masked layers,” 2020, arXiv:2005.06670.
[21] S. Han, H. Mao, and J. W. Dally, “Deep compression: Compressing deep neural network with pruning, trained quantization and Huffman coding,” in Proc. 4th Int. Conf. Learn. Represent., (ICLR), Y. Bengio and Y. LeCun, Eds. San Juan, Puerto Rico, May 2016, pp. 1–14.
[22] S. Han, J. Pool, J. Tran, and W. Dally, “Learning both weights and connections for efficient neural network,” in Proc. Adv. Neural Inf. Process. Syst., vol. 28, 2015, pp. 1135–1143.
[23] 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 Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2019, pp. 2790–2799.
[24] G. Li, X. Ma, X. Wang, L. Liu, J. Xue, and X. Feng, “Fusion-catalyzed pruning for optimizing deep learning on intelligent edge devices,” IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., vol. 39, no. 11, pp. 3614–3626, Nov. 2020.
