Neural Network Compression Using Higher-Order Statistics and Auxiliary Reconstruction Losses

Christos Chatzikonstantinou Georgios Th. Papadopoulos Kosmas Dimitropoulos Petros Daras
Information Technologies Institute, Centre for Research and Technology Hellas, Greece
{chatziko,papad,dimitrop,daras}@iti.gr

Abstract

In this paper, the problem of pruning and compressing the weights of various layers of deep neural networks is investigated. The proposed method aims to remove redundant filters from the network to reduce computational complexity and storage requirements, while improving the performance of the original network. More specifically, a novel filter selection criterion is introduced based on the fact that filters whose weights follow a Gaussian distribution correspond to hidden units that do not capture important aspects of data. To this end, Higher Order Statistics (HOS) are used and filters with low cumulant values that do not deviate significantly from Gaussian distribution are identified and removed from the network. In addition, a novel pruning strategy is proposed aiming to decide on the pruning ratio of each layer using the Shapiro-Wilk normality test. The use of auxiliary MSE losses (intermediate and after the softmax layer) during the fine-tuning phase further improves the overall performance of the compressed network. Extensive experiments with different network architectures and comparison with state-of-the-art approaches on well-known public datasets, such as CIFAR-10, CIFAR-100 and ILSCVR-12, demonstrate the great potential of the proposed approach.

1. Introduction

Deep Neural Networks (DNNs) usage has been extraordinarily expanded recently, due to the fact that such networks have been experimentally shown to provide robust solutions to numerous and diverse applications in which machine learning techniques are required, e.g., computer vision, natural language processing and speech recognition, etc [21, 22]. The remarkable success of DNNs in the aforementioned tasks has inevitably triggered significant attention towards their implementation in embedded systems, so as to enable their application in settings of decreased availability of computational resources. To this end, methods targeting the reduction of the DNN parameters, while maintaining their recognition performance, have received particular attention within the deep learning community. These approaches are typically termed DNN acceleration and compression techniques.

Multiple network acceleration and compression approaches have been proposed so far and they can be roughly grouped into five main categories: a) Pruning methods, which aim at removing redundant parameters from the network in order to reduce computational complexity and storage requirements [2, 7, 11, 45]; b) low-rank approximation methods that make use of decomposition techniques to split the DNN convolutional matrices into smaller ones in order to reduce the computational complexity of the network [10]; c) teacher-student methods that train a more compact and computationally efficient DNN (student network), using a larger DNN (teacher network) for guidance [43]; d) compact network design strategies, that construct low-complexity network architectures at the expense of a small classification performance reduction (*e.g.*, replacement of fully connected layers with global average pooling operators [37], use of depthwise separable convolutions [24], etc.); e) network quantization methods that reduce the precision of DNN parameter values enabling the reduction of storage requirements and computational complexity [44].

Pruning of filters' parameters is one of the most popular compression approaches, which are based on the assumption that many parameters in DNNs are often redundant. Depending on the part of the network that they operate, pruning methods are categorized into different categories [3, 5]. Filter-level pruning techniques [7], which aim at removing convolutional filters or feature channels, have received particular attention due to their reported efficiency in compressing DNNs and their reduced implementation simplicity. Nevertheless, no attention has been drawn to the higher-order statistical characteristics of the DNN parameters. Thus, the existing filter-level pruning approaches do not take into consideration the Gaussian priors, which are imposed to the DNN parameters during training or ini-

tialization

The fundamental consideration of the current work is based on the observation that DNN training initialization (and hence DNN learning tendency) typically forces filters parameters to follow a Gaussian distribution resulting in hidden units with negligible effect to the network output [29]. In this respect, the values of the DNN filter parameters can be considered as signal samples generated by a zero mean stationary process. Higher Order Statistics (HOS) [28], which are a particular type of cumulant metrics, provide a reliable measure of the distance of a random process from Gaussianity.

More specifically, in this paper, a four-step compression algorithm is proposed (Figure 1). Firstly, HOS are used as a filter selection criterion, where filters with low cumulant value (i.e., not deviating significantly from the Gaussian distribution) are removed. Secondly, Higher Order Orthogonal Iteration (HOOI) [19] is applied to the DNN matrices in order to reduce the computational complexity of the network. The third step involves the fine-tuning of the compressed DNN. Auxiliary mean squared error (MSE) losses are computed during the fine-tuning of the compressed DNN both after the softmax layer and the intermediate layers. Finally, the fourth step of the algorithm includes further fine-tuning of the DNN without the usage of auxiliary losses in order to improve the overall accuracy of the network. More specifically, the main contributions of this paper are summarized as follows:

- A novel filter selection criterion is proposed based on the fact that filters whose weights follow a Gaussian distribution correspond to hidden units that do not capture important aspects of data. To this end, higher-order statistics are used for the identification and removal of filters with low cumulant values, i.e., filters whose parameters deviate significantly from a Gaussian distribution.

- A novel pruning strategy is proposed aiming to define the pruning ratio of each individual layer using the Shapiro-Wilk normality test. The proposed strategy defines an adaptive per layer pruning ratio, instead of a global one, to further optimize the pruning of filters in each layer.

- To improve the recognition accuracy of the network, the employment of auxiliary MSE losses (intermediate and after the softmax layer) during the fine-tuning step is proposed. Experimental results show that the auxiliary losses optimize the filter parameters improving the performance of the compressed network.

Extensive experiments with different network architectures and comparison with state-of-the-art approaches on well-known public datasets, such as CIFAR-10, CIFAR-100 and ILSCVR-12, demonstrate the efficiency of the proposed approach.

The remainder of the paper is organized as follows: Relevant previous work is discussed in Section 2. The proposed method is presented in Section 3. Thorough experimental analysis and extensive comparative evaluation are provided in Section 4. Finally, conclusions are drawn in Section 5.

2. Related Work

Filter-level pruning methods target the removal of convolutional filters or feature channels, in order to reduce the DNN parameter space. One of the main advantages of these methods is their compression efficiency (while maintaining high classification accuracy), combined with increased implementation simplicity. More specifically, Li *et al*. [23] prune a certain percentage of channels in each layer, based on the L1-norm of its filter weights, while in [13] a filter-pruning approach is introduced, which prunes the filters with a median value similar to the median value of the filters within the same layer. Moreover in [26] L1 regularization is imposed to the importance factor of each layer during training and, subsequently, channels with lower importance are discarded. Yu *et al*. [42] introduce a so called Final Response Layer (FRL) and neurons in previous layers are removed based on the propagation of the importance scores to the FRL one. On the other hand, Ding *et al*. [7] propose a SGD optimization method, which creates identical filters during training. All but one identical filters are pruned with minimum accuracy loss. Additionally, You *et al*. [41] multiply the output of each filter with a trainable parameter, taking advantage of the Taylor expansion to estimate the change in the loss function. The less important filters are pruned based on the modification of the loss function. Neural architecture search is used in [8] to define the channel and layer sizes of the pruned network. Furthermore, Lin *et al*. [25], solve the optimization problem of network pruning using a learning algorithm inspired by Generative Adversarial Networks (GANs).

On the other hand, some filter-level pruning methods take advantage of the MSE loss to guide the pruning process. Luo *et al*. [14] and He *et al*. [27] prune the filters which have the smallest impact to the reconstruction error between the baseline and the pruned network. In [45] auxiliary losses are added into the network to increase the discriminative power of intermediate layers. Channel selection is conducted based on the additional losses and the reconstruction error of the feature maps. Contrary to previous methods, which used the MSE loss for channel selection, in the proposed method the MSE loss is employed during the fine-tuning step of the compression algorithm.

Common ground of the existing filter-level pruning methods is that they do not investigate the higher-order statistical characteristics of the DNN parameters before apply-

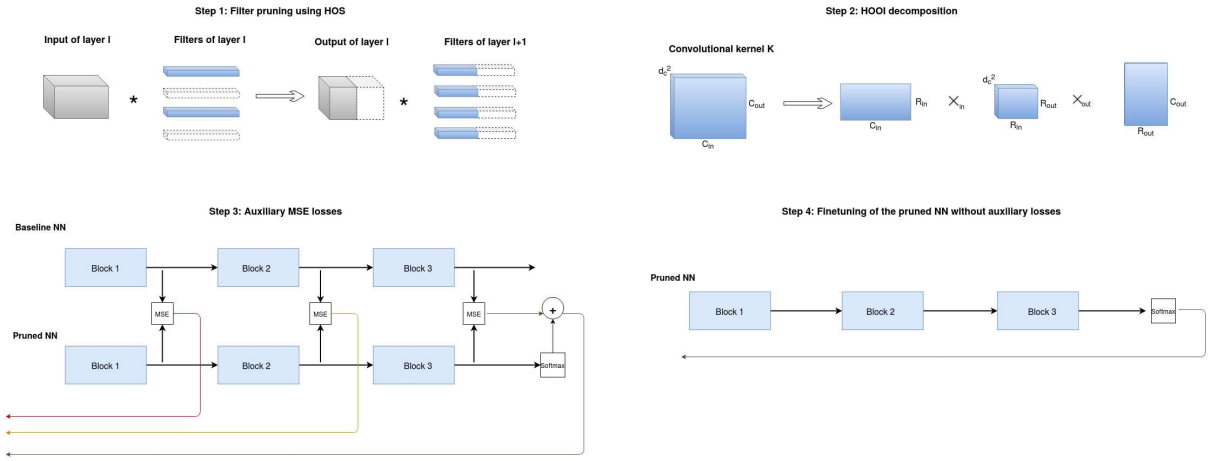

Figure 1. Outline of the proposed method. Initially, the HOS metrics are calculated for the filters of layer L and the least discriminant filters are removed. Corresponding layer outputs and kernels in the subsequent layer are also pruned. Afterwards, the HOOI algorithm is applied to the convolutional kernels. The initial convolutional kerners are replaced with three smaller ones. The next step is the fine-tuning of the DNN, employing auxiliary MSE losses at the output of the network and after some intermediate layers. Finally, the DNN is fine-tuned without the usage of the aforementioned losses.

ing pruning operations. The fact that Gaussian priors are typically imposed to the DNN parameters, either during initialization or due to the employment of the L2 norm for regularization during training, is not taken into account. In this work, the fact that the DNN filter parameters are typically forced to follow a Gaussian distribution is utilized in order to prune the filters with negligible effect to the DNN output.

3. Proposed approach

3.1. Motivation

It is common practice in the deep learning community to use the L2 norm for regularization during training, in an attempt to address the problem of exploding gradients [30]. However, L2 regularization forces the DNN filter parameters to follow a Gaussian distribution [4], [33]. Nevertheless, Gaussianity is not an ideal characteristic for DNNs. Specifically, "...with Gaussian priors the contributions of individual hidden units are all negligible, and consequently, these units do not represent "hidden features" that capture important aspects of the data..." [29]. Suitable metrics, originating from the information theory field, for measuring the distance of a random process from Gaussianity are HOS. To this end, HOS are utilized in this work as a criterion for detecting (and subsequently removing) the convolutional filters that do not capture important and discriminative aspects of the data.

3.2. Higher Order Statistics (HOS)

HOS [36] are defined in terms of moments and cumulants. For a zero mean stationary random process $z(t)$, where $(\tau_1, \tau_2, ..., \tau_{q-1})$ are the $q-1$ time lags, and, for $q = 3, 4$, the q^{th} order cumulant of $z(t)$ can be defined as the difference between $z(t)$ and $g(t)$ [$g(t)$ is a Gaussian random process with the same second-order statistics as $z(t)$]:

$$C_{q,z}(\tau_1, \tau_2, ..., \tau_{q-1}) = E\{z(\tau_1), ..., z(\tau_{q-1})\} - E\{g(\tau_1), ..., g(\tau_{q-1})\}, \quad (1)$$

where $C_{q,z}$ is the q^{th} order cumulant of $z(t)$. Consequently, cumulants can provide a measure of the distance of a random process from Gaussianity.

Cumulants of a set of values with sample size N can be calculated using k-order statistics. The q^{th} order k-statistic is the unbiased estimator of the cumulant $C_{q,z}$. In this respect, the 3rd and 4th order statistics can be computed, using the central moments m_k, as follows [17]:

$$k_3 = \frac{N^2}{(N-1)\cdot(N-2)} m_3, \quad (2)$$

$$k_4 = \frac{N^2[(N+1)m_4 - 3(N-1)m_2^2]}{(N-1)\cdot(N-2)\cdot(N-3)} \quad (3)$$

Regarding the most commonly met HOS, skewness (γ_1) and its commonly used estimator (denoted g_1) are defined, using the raw central moments μ_p, according to the following expressions [1]:

$$\gamma_1 = \frac{\mu_3}{\mu_2^{\frac{3}{2}}}, \quad (4)$$

$$g_1 = \frac{k_3}{k_2^{\frac{3}{2}}}. \quad (5)$$

On the other hand, kurtosis is defined as follows [1]:

$$\beta_2 = \frac{\mu_4}{\mu_2^2} \quad (6)$$

The most commonly used form though is kurtosis excess (γ_2) and its estimator (denoted g_2) [1]:

$$\gamma_2 = \frac{\mu_4}{\mu_2^2} - 3 \tag{7}$$

$$g_2 = \frac{k_4}{k_2^2} \tag{8}$$

3.3. Selection of pruning filters

According to the proposed approach, a filter-level pruning method is introduced. This method makes use of HOS values (Section 3.2) as a filter selection criterion. In particular, the values of the DNN filter parameters can be considered as signal samples generated by a zero mean stationary process as it can be reasonably assumed based on the observation that L2 norm forces the DNN filter parameters to follow a Gaussian distribution. The aforementioned assumption has also been experimentally evaluated, by calculating the mean value of each filter's parameters and making use of the augmented Dickey-Fuller test for evaluating the stationarity aspect [6]. More specifically, the following HOS metrics are studied in this work:

- The 3rd order cumulant, k_3 (Eq. 2).
- The 4th order cumulant, k_4 (Eq. 3).
- The product of k_3 and k_4.
- The estimator of skewness, g_1 (Eq. 4).
- The estimator of kurtosis excess, g_2 (Eq. 7).
- The product of g_1 and g_2.

Concering the pruning process, a pretrained DNN is pruned in a single step and subsequently it is fine-tuned. Filters whose weight distribution exhibits a small distance from the Gaussian one are discarded since they have negligible impact to the overall DNN outcome.

3.4. Pruning strategy

The proposed pruning method is applied to the filters of a DNN, as shown in (Figure 1). Initially, the higher order cumulants of the filters of each layer are calculated. Then, the filters are pruned based on their distance from the Gaussian distribution. Subsequently, corresponding layer outputs and kernels in the subsequent layer are also removed. A critical aspect in DNN compression concerns the pruning ratio applied to each layer.

In the current work, a novel filter-pruning strategy is proposed, where the pruning process is guided by the metric of a normality test. The latter evaluates the validity of the null hypothesis that a sample originates from a normally distributed population. The test is applied to each network layer and the resulting value of the test statistic (T) is used to define the pruning ratio of each layer. A higher value of T indicates that the sample is more likely to originate from a Gaussian distribution. Given the fact that a Gaussian distribution of the DNN parameters is not desired (as already discussed in the beginning of this section), a layer exhibiting a higher T value is associated with a greater pruning ratio. The pruning ratio R of each layer is calculated according to the following expression:

$$R = \frac{T - T_{min}}{coef \cdot (T_{max} - T_{min})}, \tag{9}$$

where T_{max} is the maximum T value estimated for the DNN, T_{min} is the corresponding minimum T value and $coef$ is a coefficient that varies depending on the targeted pruning ratio.

A per layer pruning ratio is estimated, instead of defining a global NN filter-pruning threshold, for ensuring that no layers end up with all the filters pruned. The Shapiro-Wilk criterion [39] was selected as a formal normality test among others due to its increased robustness [32], which was also evaluated experimentally (Section 4.3.3).

3.5. Higher Order Orthogonal Iteration (HOOI)

At the second step of the proposed compression algorithm, the HOOI [19] algorithm is applied to the pruned convolutional kernels so as to achieve a further reduction of the DNN parameters, through the estimation of a low-rank approxiamtion of the network tensors. The HOOI algorithm consists of two steps: the Higher Order SVD (HOSVD) analysis and the Alternating Least Squares (ALS) algorithm. The HOSVD is a method used to calculate the Tucker decomposition of a N-dimensional tensor. It is used as an initial step for the first estimation of the Tucker decomposition and subsequently the ALS algorithm is used to better estimate the decomposed matrices.

The initial convolutional kernel $K \in \mathbb{R}^{w \cdot h \cdot C_{in} \cdot C_{out}}$ is replaced with three smaller convolutional kernels, $K_C \in \mathbb{R}^{d_C \cdot d_C \cdot C_{in} \cdot C_{out}}$, $K_{in} \in \mathbb{R}^{d_{in} \cdot d_{in} \cdot C_{in} \cdot R_{in}}$ and $K_{out} \in \mathbb{R}^{d_{out} \cdot d_{out} \cdot R_{out} \cdot C_{out}}$, with C_{in} input filters, C_{out} output filters, w, h the spatial dimensions of the filter, $d_C = 3$ and $d_{out} = d_{in} = 1$.

Specifically, the layer output $Y \in \mathbb{R}^{W_y \cdot H_y \cdot C_{out}}$, where W_y and H_y are the spatial dimensions of the output feature map, arises from the successive convolutions of the layer input with the tensors resulting from the low-rank approximation of the initial one [10]:

$$O_1 = K_{in} \circledast X, \;\; O_2 = K_C \circledast O_1, \;\; Y = K_{out} \circledast O_2, \tag{10}$$

where $X \in \mathbb{R}^{W_x \cdot H_x \cdot C_{in}}$, with W_x and H_x the spatial dimensions of the input feature map. The ranks of each network layer are decided by employing a constant compression rate.

3.6. Auxiliary MSE losses

Most existing methods include a fine-tuning step after the pruning of the initial network. In this method, a mean squared error (MSE) term is inserted after the softmax layer as an auxiliary loss in order to improve the convergence of the pruned network. The MSE loss can be defined as:

$$L_{MSE} = \frac{1}{N} \cdot (S^p - S^b)^2, \tag{11}$$

where S^p denotes the softmax output of the pruned network, S^b is the softmax output of the baseline network and $N = C_{in}{*}C_{out}$. The final loss of the DNN is:

$$L_f = L_s + R * L_{MSE}, \tag{12}$$

where L_s is the softmax layer and R is a factor which weights the impact of the MSE loss to the final loss.

The depth of CNNs reduces the discriminative power of intermediate layers due to the long backpropagation path. To this end, weighted MSE auxiliary losses are introduced to intermediate convolutional layers and can be defined as:

$$L_{MSE_c} = \frac{1}{N_c} \cdot \sum_{i=1}^{W_y} \cdot \sum_{j=1}^{H_y} (O^p_{i,j,:,:} - O^b_{i,j,:,:})^2, \tag{13}$$

where $N_c = C_{in} * C_{out} * W_y * H_y$, O^p denotes the convolutional output of the pruned network and O^b is the convolutional output of the baseline network. Moreover, the gradients of L_{MSE_c} are multiplied by a factor:

$$f = max(0.01, (\frac{L_k}{L_K})^\nu) \text{ [9]}, \tag{14}$$

where $\nu > 0$ is the decaying rate of f, L_k is the index of the layer the loss is applied and L_K is the index of the softmax layer.

Zhuang *et al*. [45] also use auxiliary losses to intermediate convolutional layers. However, in contrast to the proposed approach, the auxiliary losses are softmax losses instead of MSE losses. Moreover, those losses are utilised during the channel selection step rather than during the fine-tuning step.

The last step of the proposed algorithm involves some epochs of fine-tuning, ommiting the MSE losses and exploring the ability of the pruned DNN to achieve an increased accuracy compared to the baseline one.

4. Experimental results

4.1. Datasets

For the experimental evaluation, the following well-known public datasets are used for the task of image classification:

CIFAR: The CIFAR datasets [20] consist of natural images with resolution 32x32 that belong to 10 semantic classes in the case of the CIFAR-10 dataset and 100 semantic classes in the case of the CIFAR-100 dataset. The training and test sets contain 50K and 10K images, respectively.

ILSCVR-12: The ILSCVR-12 (ImageNet Large Scale Visual Recognition Challenge-2012) [35] includes approximately 1.2M training and 50K test images belonging to 1000 object categories.

4.2. Implementation details

Regarding implementation details, the input data are normalized using each channel's mean and standard deviation values. A pre-defined data augmentation scheme is adopted [12], which employs random cropping and horizontal flip operations. The training set is split to training and validation set. Regarding the CIFAR datasets, the training set consists of the 90% of the images of the default training set and the validation set of the remaining 10%, whereas regarding the ILSCVR-12, the training set consists of the 95% of the images of the default training set and the validation set of the remaining 5%.

With respect to the implemented DNNs, the PyTorch framework [31] is used for all development activities. The proposed approach is evaluated using two popular network architectures, namely ResNet [12] and MobileNet [15] [38]. In all cases, the Stochastic Gradient Descent (SGD) optimizer [34] is used with Nesterov momentum [40] equal to 0.9 and batch size set to 64.

The baseline DNNs are trained for 200 epochs using the CIFAR-10 dataset, with learning rate 0.1, divided by 10 at 100 and 150 epochs. Concerning the CIFAR-100 dataset, the baseline DNNs are trained for 300 epochs using learning rate 0.1, which is divided by 10 at 150 and 225 epochs. Regarding the ILSCVR-12 dataset, the pretrained PyTorch baseline networks are used.

CIFAR training pipeline: The weights of the pre-trained networks are updated for 40 epochs, with learning rate set equal to 0.001. From the experiments of Section 4.3.3 onwards, which employ bigger pruning ratios, the pretrained DNNs are fine-tuned for 140 epochs with learning rate equal to 0.001, which is divided by 10 at 80 and 120 epochs. Concerning the MobileNet networks, they are trained for 90 epochs, the learning rate is equal to 0.001 and it is divided by 10 at 60 epochs.

ILSCVR training pipeline: The weights of the pretrained networks are updated for 70 epochs, with learning rate set equal to 0.001, divided by 10 at epochs 10 and 50.

Regarding the pruning strategies, for the ResNet architecture, the last layer of each residual block is not pruned, guided by the respective common practice in the literature [23, 27]. Additionally, the implemented pruning strate-

gies target the selection of network filters; therefore, the fully connected layers are not pruned. As for the depthwise separable convolutional layers of the Mobilenets, they are not directly pruned. Specifically, if the previous layer is pruned, the respective filters of the depthwise separable layer are pruned too. Concerning the low-rank approximation method, every layer of the network is decomposed (fully connected and convolutional) apart from the first convolutional layer and the softmax layer, which cause a severe deterioration of the DNN performance when decomposed.

Code is available at: `https://github.com/chatzikon/DNN-COMPRESSION`.

4.3. Experimental evaluation

4.3.1 Evaluation of pruning criteria

The main goal of the first set of the conducted experiments is to assess the efficiency of the six proposed filter selection criteria, namely the HOS metrics introduced in Section 3.3. Furthermore, the HOS metrics are compared with simple criteria, such as L1-norm and random filter selection. Extensive experimental evaluation is performed and the obtained results on CIFAR-10 dataset are summarized in Table 1. For all cases, the difference of the obtained classification accuracy of the compressed DNN is compared with the one of the original DNN.

In order to evaluate the efficiency of the introduced HOS metrics for filter selection, the proposed pruning strategy is used employing the Shapiro-Wilk normality test. Based on the experimental results in Table 1, it can be seen that k_3, k_4 and their product perform better than γ_1, γ_2 and their product for the evaluated DNNs. The latter implies that the central moments m_s are more efficient in detecting convolutional filters that bear less discriminative power compared to the respective μ_p central moments. Additionally, it can be seen that the product $k_3 \times k_4$ leads to improved recognition performance compared to that when using k_3 or k_4 alone. This indicates the merits of comparing different HOS metrics for pruning DNN filters.

Metric	Parameters pruned (%)	Recognition accuracy (%) (Top-1)
ResNet56 (Baseline accuracy: 92.64%)		
k_3	29.58	-0.99
k_4	29.58	-0.99
$\mathbf{k_3 x k_4}$	**29.58**	$\mathbf{-0.91}$
γ_1	29.58	-1.09
γ_2	29.58	-1.10
$\gamma_1 x \gamma_2$	29.58	-0.95
$L1$	29.58	-1.02
$Random$	29.58	-1.21
ResNet110 (Baseline accuracy: 93.22%)		
k_3	32.64	-0.40
k_4	32.64	-0.29
$\mathbf{k_3 x k_4}$	**32.64**	$\mathbf{-0.15}$
γ_1	32.64	-0.37
γ_2	32.64	-0.59
$\gamma_1 x \gamma_2$	32.64	-0.54
$L1$	32.64	-0.49
$Random$	32.64	-0.55

Table 1. Evaluation of different pruning metrics on CIFAR-10 dataset ($L1$ is the L1-norm and $Random$ is the random filter selection).

Pruning strategy (%)	HOS metric	Parameters pruned	Recognition accuracy (%) (Top-1)
ResNet56 (Baseline accuracy: 92.64)			
P1	$\mathbf{k_3 x k_4}$	**29.58**	$\mathbf{-0.91}$
P2	$k_3 x k_4$	29.59	-0.95
P3	$k_3 x k_4$	29.59	-0.95
P4	$k_3 x k_4$	29.62	-1.45
P5	$k_3 x k_4$	29.62	-0.98
ResNet110 (Baseline accuracy: 93.22)			
P1	$\mathbf{k_3 x k_4}$	**32.64**	$\mathbf{-0.15}$
P2	$k_3 x k_4$	32.3	-0.47
P3	$k_3 x k_4$	32.5	-0.52
P4	$k_3 x k_4$	32.19	-0.67
P5	$k_3 x k_4$	32.4	-0.17

Table 2. Evaluation of different pruning strategies on CIFAR-10 dataset.

4.3.2 Evaluation of pruning strategies

In this section, the proposed DNN pruning strategy (described in Section 3.4) is evaluated. More specifically, the normality test used in P1 pruning strategy is Shapiro-Wilk, while Jarque-Bera test [16] is used in P2 pruning strategy. On the other hand, in P3 the $k_3 \times k_4$ metric is used instead of a normality test. In P4 pruning strategy, a global pruning threshold is used, whereas in P5 the pruning strategy proposed in Li *et al*. [23] is employed. Quantitative evaluation results are given for each pruning strategy and for different DNN architectures. From the experimental results in Table 1, it can be seen that the proposed pruning strategy (P1) outperforms all other approaches both in ResNet56 and ResNet110 networks.

4.3.3 Evaluation of different compression approaches

In this section, we evaluate three different compression approaches. More specifically, in the first approach (A1) only the filter-level pruning method is applied to the network, whereas in A2 approach, the network is compressed using exclusively the low-rank approximation (i.e., filter pruning

Compression approach (%)	Parameters pruned (%)	Parameters removed	Recognition accuracy (%) (Top-1)
ResNet56 (Baseline accuracy: 92.64)			
A1	70.97	70.97	-2.35
A2	–	70.69	-1.71
A3	**29.58**	**70.37**	$\mathbf{-1.58}$
ResNet110 (Baseline accuracy: 93.22)			
A1	61.39	61.39	-1.34
A2	–	62.09	-1.02
A3	**32.64**	**61.36**	$\mathbf{-0.83}$

Table 3. Evaluation of different compression approaches on CIFAR-10 dataset.

Network	Factor R	No of intermediate MSE losses	Decay rate ν	Recognition accuracy (%) (Top-1)
ResNet56 (Baseline accuracy: 92.64)				
DNN without MSE	-	-	-	-1.58
DNN with MSE	**5**	**2**	**3**	$\mathbf{+0.08}$
ResNet110 (Baseline accuracy: 93.22)				
DNN without MSE	-	-	-	-0.83
DNN with MSE	**2**	**2**	**4**	$\mathbf{+0.31}$

Table 4. Experiments using the auxiliary MSE losses and the ResNet architecture on CIFAR-10 dataset.

is not applied in this case). The A3 approach is a combination of A1 and A2 in which the network is initially pruned and then it is further compressed using the HOOI algorithm. The experimental results with the three different pruning strategies are presented in Table 3. The third column of the table reports the total removed parameters of the DNN due to either the filter-pruning or the low-rank approximation or both. In all experimental results presented in Table 3, the best performing pruning metric, i.e., $k_3 \times k_4$ is used, while pruning strategy P1 is adopted in the case of A1 and A3 compression approaches.

From the experimental results presented in Table 3, it can be seen that A1 compression approach suffers from the largest recognition accuracy drop (compared with the recognition accuracy of the original DNN), i.e., -2.35 and -1.34 in the case of ResNet56 and ResNet110, respectively. On the other hand, low rank approximation approach, i.e., A2, provides better recognition accuracy with a drop of -1.71 and -1.02 in ResNet56 and ResNet110, respectively. However, the combination of filter-level pruning and low-rank approximation (i.e., A3 compression approach) improves the experimental results in terms of compression (parameters removed) and recognition accuracy in both networks revealing the great potential of the proposed compression approach.

4.3.4 Evaluation of the auxiliary MSE losses

The current section examines the effect of the auxiliary MSE losses during fine-tuning. Three different hyperparameters are tuned: the R factor of the Eq. 12, the number of the intermediate auxiliary MSE losses and the value of ν (Eq. 14). The best combinations of the above parameters are presented in Table 4, along with the achieved recognition accuracy. In all experiments, the metric k_3xk_4, the pruning strategy P1 and the compression approach A3 are used. From the experimental results in Table 4, it can be easily seen that the use of MSE losses during fine-tuning phase improves significantly the recognition accuracy of the network. In both cases, i.e., ResNet56 and ResNet110, the recognition accuracy of the compressed network is higher than that of the baseline network.

4.3.5 Comparison against state-of-the-art literature approaches

The proposed method is also comparatively evaluated against state-of-the-art filter-level DNN pruning methods. In particular, the currently best performing methods in the literature [7, 8, 13, 25, 41, 42, 45] are included in this study. The obtained comparative evaluation results on CIFAR-10, CIFAR-100 and ILSCVR-12 datasets are presented in Tables 5, 6 and 7, respectively. It needs to be noted that for the proposed approach the best performing pruning metric (namely k_3xk_4), and pruning strategy (P1) are used, as experimentally verified in previous sections. From the presented results, it can be seen that the proposed approach achieves superior performance. The latter demonstates the added value of coupling the use of HOS metrics for filter selection with the introduced statistical analysis-based pruning strategy and the HOOI algorithm.

More specifically, the proposed method increases the Top-1 recognition accuracy, compared with the one of the baseline DNN, on CIFAR-10 dataset (Table 5) by 0.08, 0.31, 0.56 and 0.82 in the cases of ResNet56, ResNet110, MobileNetV1 and MobileNetV2, respectively. Moreover, concerning the CIFAR-100 dataset (Table 6), the Top-1 recognition accuracy of ResNet56 is slightly decreased by 0.01, whereas the Top-1 recognition accuracy of ResNet110 is increased by 0.12. Finally, we can see in Table 7 that the Top-1 and Top-5 recognition accuracies of MobileNetV2 on ILSCVR-12 dataset are decreased by 5.68 and 2.87, respectively.

4.3.6 Reduction of the model size

In this section, the reduction rate of the parameters removed from the initial DNN is compared to the reduction of the model size of the pruned networks in order to find out the storage benefit of the proposed method. The results can be seen in Table 8. The models compressed in the previous

Method	Parameters removed (%)	FLOPs removed (%)	Recognition accuracy (%) (Top-1)
ResNet56			
Yu *et al*. [42]	43.61	42.6	−0.03
Lin *et al*. [25]	65.9	60.2	−1.68
Dong *et al*. [8]	–	52.7	−0.77
He *et al*. [13]	–	52.6	−0.10
Ding *et al*. [7]	–	60.85	+0.05
Zhuang *et al*. [45]	70.33	47.09	+0.01
You *et al*. [41]	66.7	70.3	−0.03
Proposed	**70.37**	**69.72**	**+0.08**
ResNet110			
Lin *et al*. [25]	44.8	48.5	−0.76
Dong *et al*. [8]	–	53	−0.64
He *et al*. [13]	–	52.3	+0.17
Ding *et al*. [7]	–	60.89	+0.06
Proposed	**61.36**	**61.04**	**+0.31**
MobilenetV1			
Zhuang *et al*. [45]	30.07	42.86	+0.41
Proposed	**30.55**	**38.89**	**+0.56**
MobilenetV2			
Zhuang *et al*. [45]	23.66	26.47	+0.22
Proposed	**23.69**	**26.82**	**+0.82**

Table 5. Comparative evaluation results using the CIFAR-10 dataset.

Method	Parameters removed (%)	FLOPs removed (%)	Recognition accuracy (%) (Top-1)
ResNet56			
Dong *et al*. [8]	–	51.3	−0.93
Proposed	**50.32**	**51.54**	**−0.01**
ResNet110			
Dong *et al*. [8]	–	52.6	−1.9
Proposed	**54.36**	**53.87**	**+0.12**

Table 6. Comparative evaluation results using the CIFAR-100 dataset.

Method	Parameters pruned (%)	FLOPs removed (%)	Recognition accuracy (%) (Top-1/Top-5)
Zhuang *et al*. [45]	25.93	44.75	−5.89/ − 3.77
Proposed	**27.13**	**43.65**	**−5.68/−2.87**

Table 7. Comparative evaluation results using the ILSCVR-12 dataset (and MobileNetV2).

Network	Parameters removed (%)	Model size reduction (%)
ResNet56	70.37	68.57
ResNet110	61.36	59.15
MobileNetV1	30.55	30.5
MobileNetV2	23.69	23.41

Table 8. Comparison between the reduction rate of the model parameters and the model size reduction on CIFAR-10 dataset

sections are examined and the size reduction of the models trained using CIFAR-10 dataset is presented. Based on the results of Table 8, it can be concluded that the reduction rate of the model parameters is very similar to the model size reduction, especially in the case of the MobileNet networks.

Network	FLOPs removed (%)	Inference time (s)	Inference memory (MB)	FPS at a smart device
MobileNetV2	0	138	2561	7
Compressed MobileNetV2	**43.65**	**138**	**1961**	**8**

Table 9. Comparison between the inference memory and the inference time required at a desktop computer and the FPS required at a smart device by the compressed model and the baseline one

4.3.7 Integration to a mobile device

In this section, the inference memory and the inference time required by the MobileNetV2, trained on ILSCVR-12, is examined. Initially, the inference time and the inference memory of the baseline and the compressed model are examined at a desktop computer, using a NVIDIA GeForce GTX 1070 GPU. As shown in Table 9, there is no difference in the execution time during inference, however the compressed model requires 23.43% less inference memory. Moreover, the baseline and the compressed model are integrated to a mobile device (MLS MX Pro) using the Pytorch demo app [18]. The compressed model achieves 12.5% speedup in terms of FPS according to the results presented in Table 9.

5. Conclusions

In this paper, the problem of filter-level pruning for realizing DNN compression was investigated. A novel filter selection methodology was proposed, which is based on the use of higher order statistics (HOS), where filters with low cumulant value (i.e., not deviating significantly from the Gaussian distribution) are discarded. A new parameter pruning strategy was introduced, which makes use of the Shapiro-Wilk normality test to decide on the pruning ratio of each individual layer. Furthermore, the Tucker decomposition is used for further compression as well as auxiliary MSE losses are employed in intermediate network layers in order to improve the performance of the pruned network. Extensive experimentation as well as comparative evaluation with similar literature approaches, involving well-known public datasets (CIFAR-10, CIFAR-100 and ILSCVR-12) and multiple network architectures, demonstrated the efficiency of the proposed approach.

Future work includes the investigation of additional alternative DNN pruning methods, which will focus not only on the statistical analysis of the DNN properties, but will also take into account the semantic information that is conveyed by the different parts of the DNN.

Acknowledgement

This work was supported by the Greek General Secretariat of Research and Technology under contract T1EK-02469 EPIKOINONO.

References

[1] M. Abramowitz and I. Stegun. *Handbook of mathematical functions*. Dover Publications Inc., 1970.

[2] Amir H Ashouri, Tarek S Abdelrahman and Alwyn Dos Remedios. Retraining-free methods for fast on-the-fly pruning of convolutional neural networks. *Neurocomputing*, 370:56–69, 2019.

[3] Davis Blalock, Jose Javier Gonzalez Ortiz, Jonathan Frankle and John Guttag. What is the state of neural network pruning? *arXiv preprint arXiv:2003.03033*, 2020.

[4] C. Blundell, J. Cornebise, K. Kavukcuoglu and D. Wierstra. Weight uncertainty in neural network. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 1613–1622. PMLR, 07–09 Jul 2015.

[5] Jian Cheng, Pei-song Wang, Gang Li, Qing-hao Hu and Hanqing Lu. Recent advances in efficient computation of deep convolutional neural networks. *Frontiers of Information Technology & Electronic Engineering*, 19(1):64–77, 2018.

[6] Yin-Wong. Cheung and Kon S. Lai. Power of the augmented dickey-fuller test with information-based lag selection. *Journal of Statistical Computation and Simulation*, 60(1):57–65, 1998.

[7] Xiaohan Ding, Guiguang Ding, Yuchen Guo and Jungong Han. Centripetal sgd for pruning very deep convolutional networks with complicated structure. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4943–4953, 2019.

[8] Xuanyi Dong and Yi Yang. Network pruning via transformable architecture search. In *Advances in Neural Information Processing Systems*, pages 759–770, 2019.

[9] Yong Guo, Mingkui Tan, Qingyao Wu, Jian Chen, Anton Van Den Hengel and Qinfeng Shi. The shallow end: Empowering shallower deep-convolutional networks through auxiliary outputs. *arXiv preprint arXiv:1611.01773*, 2016.

[10] Julia Gusak, Maksym Kholiavchenko, Evgeny Ponomarev, Larisa Markeeva, Philip Blagoveschensky, Andrzej Cichocki and Ivan Oseledets. Automated multi-stage compression of neural networks. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 0–0, 2019.

[11] Song Han, Huizi Mao and William J Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. In *International Conference on Learning Representations (ICLR)*, 2016.

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[13] Yang He, Ping Liu, Ziwei Wang, Zhilan Hu and Yi Yang. Filter pruning via geometric median for deep convolutional neural networks acceleration. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4340–4349, 2019.

[14] Yihui He, Xiangyu Zhang and Jian Sun. Channel pruning for accelerating very deep neural networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1389–1397, 2017.

[15] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

[16] Carlos M Jarque and Anil K Bera. Efficient tests for normality, homoscedasticity and serial independence of regression residuals. *Economics letters*, 6(3):255–259, 1980.

[17] J. F. Kenney and E. S. Keeping. *Mathematics of statistics, Part 1*. Van Nostrand, 1939.

[18] Ivan Kobzarev. Pytorch android examples. https://github.com/pytorch/android-demo-app.

[19] Tamara G Kolda and Brett W Bader. Tensor decompositions and applications. *SIAM review*, 51(3):455–500, 2009.

[20] A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images. Technical Report 4, University of Toronto, 2009.

[21] Yann LeCun, Yoshua Bengio and Geoffrey Hinton. Deep learning. volume 521, pages 436–444. Nature Publishing Group, 2015.

[22] Yann LeCun, Koray Kavukcuoglu and Clément Farabet. Convolutional networks and applications in vision. In *Proceedings of 2010 IEEE international symposium on circuits and systems*, pages 253–256. IEEE, 2010.

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

[24] Min Lin, Qiang Chen and Shuicheng Yan. Network in network. 2013.

[25] Shaohui Lin, Rongrong Ji, Chenqian Yan, Baochang Zhang, Liujuan Cao, Qixiang Ye, Feiyue Huang and David Doermann. Towards optimal structured cnn pruning via generative adversarial learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2790–2799, 2019.

[26] Zhuang Liu, Jianguo Li, Zhiqiang Shen, Gao Huang, Shoumeng Yan and Changshui Zhang. Learning efficient convolutional networks through network slimming. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2736–2744, 2017.

[27] Jian-Hao Luo, Jianxin Wu and Weiyao Lin. Thinet: A filter level pruning method for deep neural network compression. In *Proceedings of the IEEE international conference on computer vision*, pages 5058–5066, 2017.

[28] J. M. Mendel. Tutorial on higher-order statistics (spectra) in signal processing and system theory: theoretical results and some applications. *Proceedings of the IEEE*, 79(3):278–305, March 1991.

[29] R. M. Neal. *Bayesian Learning for Neural Networks*. Springer-Verlag, 1996.

[30] R. Pascanu, T. Mikolov and Y. Bengio. On the difficulty of training recurrent neural networks. In *Proceedings of the 30th International Conference on Machine Learning*, volume 28 of *ICML'13*, pages III–1310–III–1318. JMLR.org, 2013.

[31] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga and A. Lerer. Automatic differentiation in pytorch. In *NIPS-W*, 2017.

[32] Nornadiah Mohd Razali, Yap Bee Wah et al. Power comparisons of shapiro-wilk, kolmogorov-smirnov, lilliefors and anderson-darling tests. *Journal of statistical modeling and analytics*, 2(1):21–33, 2011.

[33] J. Rennie. On l2-norm regularization and the gaussian prior, 2003.

[34] H. Robbins and S. Monro. A stochastic approximation method. *The Annals of Mathematical Statistics*, 22(3):400–407, September 1951.

[35] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, Dec 2015.

[36] M. Sanaullah. A review of higher order statistics and spectra in communication systems. *Global Journal of Science Frontier Research*, 13(4), June 2013.

[37] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018.

[38] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018.

[39] S. S. Shapiro and M. B. Wilk. An analysis of variance test for normality (complete samples). *Biometrika*, 52(3/4):591–611, 1965.

[40] I. Sutskever, J. Martens, G. Dahl and G. Hinton. On the importance of initialization and momentum in deep learning. In *Proceedings of the 30th International Conference on Machine Learning*, volume 28 of *ICML'13*, pages III–1139–III–1147. JMLR.org, 2013.

[41] Zhonghui You, Kun Yan, Jinmian Ye, Meng Ma and Ping Wang. Gate decorator: Global filter pruning method for accelerating deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 2130–2141, 2019.

[42] R. Yu, A. Li, C. F. Chen, J. H. Lai, V. I. Morariu, X. Han, M. Gao, C. Y. Lin and L. S. Davis. Nisp: Pruning networks using neuron importance score propagation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[43] Sergey Zagoruyko and Nikos Komodakis. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. 2016.

[44] Dongqing Zhang, Jiaolong Yang, Dongqiangzi Ye and Gang Hua. Lq-nets: Learned quantization for highly accurate and compact deep neural networks. In *Proceedings of the European conference on computer vision (ECCV)*, pages 365–382, 2018.

[45] Zhuangwei Zhuang, Mingkui Tan, Bohan Zhuang, Jing Liu, Yong Guo, Qingyao Wu, Junzhou Huang and Jinhui Zhu. Discrimination-aware channel pruning for deep neural networks. In *Advances in Neural Information Processing Systems*, pages 875–886. 2018.