


Versatile kernel reactivation for deep convolutional neural networks

Jeong Jun Lee and Hyun Kim 

Department of Electrical and Information Engineering, Research Center for Electrical and Information Technology, Seoul National University of Science and Technology, Seoul, Korea

✉ Email: hyunkim@seoultech.ac.kr

Several networks exhibiting excellent performance in the field of computer vision still suffer from the challenge of over-parameterization. Over-parameterized networks have several parameters that cannot be trained, owing to the poor backpropagation process caused by a small L1 norm, and these parameters suppress the potential for network performance improvement. This study proposes a kernel reactivation method to improve network performance by reusing invalid kernels that cannot be utilized for training because of the small L1 norm. The results indicate that the accuracy of Cifar-10 in ResNet-110 is improved by 0.94% compared to the baseline, and the top-1 and top-5 accuracies of Tiny-ImageNet in ResNet-50 are improved by 1.87% and 1.03% compared to the baseline, respectively.

Introduction: Recently, with the development of GPUs and hardware accelerators, various convolutional neural networks (CNNs) have achieved excellent performance in the field of computer vision, such as classification, object detection, and segmentation [1–3]. Several studies are in progress to improve network performance, and most of these approaches are generally accompanied by an increase in network size. However, as the network size increases, the number of parameters also increases, making the network over-parameterized [4, 5]. As a result, most high-performance networks have over-parameterized problems and in general, over-parameterized networks have several invalid parameters that cannot be trained, owing to the small L1 norm [5].

A representative method for addressing the over-parameterized challenge is network pruning [4–7]. Network pruning is a network compression method that does not degrade the network performance by removing unnecessary parameters accompanied by an increase in network size. However, in an environment with sufficient computing power support without limitation of power consumption, rather than pruning that removes some invalid parameters with small L1 norm that has no effect on training, a reactivation method that improves the network performance by changing these invalid parameters to valid parameters is further required. Representative studies approached from this point of view include weight evolution (WE) [8] and filter grafting (FG) [9] methods. WE [8] improves the network performance by reactivating invalid weights (weights with low importance) with other values, and FG [9] utilizes multiple networks to reactivate the values of all filters or layers to improve the network performance. However, since weights are the same as simple constants, WE [8], which changes the value in units of weights, has a limitation in not being able to represent features. FG [9] has a limitation in that memory usage increases significantly, because FG [9] utilizes at least two networks.

In this study, we propose a reactivation method that adopts the kernel as a unit, which is the minimum structure that can extract features, and there is no disadvantage with memory increase; thus, it can achieve excellent results on the trade-off between the accuracy and hardware resource. The contributions of this paper are summarized as follows:

- By performing reactivation in kernel units, it is possible to reactivate some parameters more sensitively than in filter or layer units, and it addresses the challenge of reactivation in weight units, in which features cannot be extracted.
- The process of selecting an invalid kernel using the L1 norm is conducted in two steps to accurately select the reactivation target. (1) The process of selecting invalid filters by considering the model and layer as a hybrid and (2) the process of adaptively selecting invalid kernels in the selected invalid filters.

- We propose a reactivation method that can preserve the features extracted from the kernel by scaling unnecessary kernels with a low L1 norm while maintaining the ratio of weights in the kernel.

Background:

Pruning: Pruning methods are divided into three categories. First, weight pruning [5] compresses the network by making unnecessary weights to zero. This method has the best compression ratio; however, it is difficult to achieve the actual acceleration effect because the network structure does not change. Second, filter pruning [4, 6] compresses the network by removing the unnecessary filters. Although this method has the best effect of accelerating the inference speed owing to structural changes, the compression ratio is relatively low. Third, kernel pruning [7] using a kernel, intermediate unit between the weight and filter, achieves excellent performance in both compression ratio and acceleration effect. In addition, because pruning is performed with a kernel, which is the smallest unit from which features can be extracted, it is the most sensitive method for maintaining features.

Reactivation: Parameters with a small L1 norm do not perform well in backpropagation during training and consequently do not significantly affect the performance of CNNs. In pruning methods, the network is compressed by removing these parameters; however, in an environment where network performance is considered more important than network compression, a reactivation method is required to use them again during training. By reactivating invalid weights, kernels, and filters that had little effect on the training process, the network performance can be improved while maintaining the size of the actual network. WE [8], which performs reactivation in units of weight, selects an invalid filter based on the L1 norm, and then conducts reactivation by replacing only one weight in all kernels of the invalid filter with the largest weight value in all kernels in the most valid filter. However, because the minimum structure that can extract the feature is the kernel, weight, which is a simple constant value, cannot extract features. Specifically, WE [8] has a limitation, in that only one weight changes for each kernel; therefore, the features that the kernel can extract are collapsed and unstable. FG [9] reactivates all layers or filters based on entropy while training several networks simultaneously. However, this method has the limitation of unstable training if the values of all filters or layers are changed simultaneously, and there is a risk that a valid parameter changes. In addition, because FG [9] must use at least two or more networks together, the memory utilized for training is required at least twice and the training time is also lengthened.

Proposed method: CNNs comprise a set of layers, where a layer is a set of filters, filter is a set of kernels, and kernel is a set of weights. Although various units can be used as units of reactivation, this study proposes a kernel unit reactivation method rather than the filter and layer units. Because the kernel is the smallest structure from which features can be extracted, it is possible to reactivate the invalid parameters sensitively, compared to the filter and layer units; therefore, it can be applied to more parts and prevent feature collapse. This study uses the L1 norm criterion, which is simple but widely utilized in pruning [10] to select an invalid kernel for kernel reactivation. Based on the L1 norm, invalid filters that satisfy two conditions (i.e., C_1 and C_2 presented below) are selected, and invalid kernels in the selected invalid filters are adaptively selected for reactivation by C_3 below. This hybrid invalid kernel selection has the advantage of stable training and there is no increase in memory because there is no additional network deployment. After selecting an invalid kernel, reactivation is performed for each certain epoch.

Condition 1 (C_1): Importance of the filter in models: First, an invalid filter is selected based on the L1 norm for kernel reactivation. Because a small L1 norm parameter has little effect on the next layer or the final network output, the L1 norm deserves the highest priority given to the conditions for reactivation. Thus, the average L1 norm of all filters in the model is calculated and sorted, and filters with the L1 norm below the hyperparameter threshold $\alpha(\%)$ are selected as invalid filter candidates.

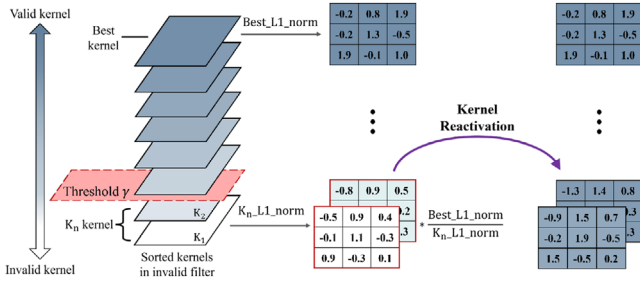


Fig. 1 Overview of kernel reactivation that preserves the ratio of the weight constituting the kernel

Condition 2 (C_2): Importance of the filter in layers: If invalid filter candidates selected through C_1 are concentrated in a specific layer, a significant portion of the corresponding layer can be reactivated. However, if several parts of the layer are altered simultaneously, the training becomes unstable and there is a risk of severe performance degradation. Therefore, when choosing the invalid filter, it is necessary to choose carefully so that the invalid filter is not too crowded in a few layers. In the proposed method, the influence of the filter in the layer is used as a second condition C_2 to select an invalid filter by also measuring them based on the L1 norm, and these filters are sorted in the order of the small L1 norm measured at each convolutional layer. The sorted filter below the hyperparameter threshold β (%) is chosen as the secondary invalid filter candidate group and a filter that satisfies both C_1 and C_2 is selected as the final invalid filter.

Condition 3 (C_3): Importance of the kernel in invalid filters: In the proposed method, all kernels in the final invalid filter selected through C_1 and C_2 are sorted by the L1 norm, and kernels with an L1 norm smaller than the hyperparameter threshold γ (%) are selected as invalid kernels that have little effect on training. The selected invalid kernels can be utilized for training again through reactivation at certain epochs, and they may have more appropriate values than existing invalid values during the remaining training process. However, because the learning rate decreases as training progresses, there is a challenge that it becomes difficult to determine more appropriate values for an invalid kernel that has conducted reactivation in a state where the learning rate is already low, which leads to deterioration of network performance. Moreover, if the γ is the same in all training processes, there is a risk that an invalid kernel at the end of training with the low learning rate may cause degradation of the model's performance after reactivation. To address these problems, in this study, we adopt the cosine annealing learning rate [11] to adaptively set the γ based on the current learning rate, and the γ is also designed to decrease in the form of a cosine graph along with the learning rate as training proceeds. Finally, the invalid kernel, K_{inv} , that performs reactivation can be expressed as follows:

$$K_{inv} = \{k | k = C_1 \cap C_2 \cap C_3\}. \quad (1)$$

All hyperparameters (α , β , γ) should be adaptively set, considering the dataset and network sizes.

Kernel reactivation: Except the 1×1 kernel, consist of several weights and become the smallest structure from which features can be extracted. To make an invalid kernel valid by reactivation, the average L1 norm of the invalid kernel needs to be increased. In addition, to avoid collapsing the features extracted by the kernel, it is also important to proceed with reactivation while maintaining the ratio of each weight constituting the kernel. Figure 1 illustrates the proposed kernel reactivation method. We denote 'K_n kernel' to the n invalid kernels in each invalid filter and 'Best kernel' to the kernel with the largest L1 norm in the filter containing the invalid kernels. 'K_n_L1_norm' and 'Best_L1_norm' denote the L1 norm of 'K_n kernel' and 'Best kernel', respectively. While maintaining the weight ratio in the kernel (i.e., the ratio in the 3×3 matrix in Fig-

Table 1. Accuracy of Cifar-10 according to β

| Network | ResNet-32 | | | | | |
|-------------------|-----------|-------|--------------|-------|-------|-------|
| Threshold β | 0.01 | 0.1 | 0.3 | 0.5 | 0.7 | 0.9 |
| Acc.(%) | 93.48 | 93.78 | 93.91 | 93.85 | 93.72 | 93.55 |

Table 2. Accuracy of Cifar-10 according to α and γ

| Network | Metric | Threshold α / γ | | | | |
|------------|---------|-----------------------------|----------|--------------|---------|--------------|
| | | Baseline | 0.1/0.05 | 0.2/0.05 | 0.1/0.1 | 0.2/0.1 |
| ResNet-32 | Acc.(%) | 93.30 | 93.82 | 93.91 | 93.80 | 93.76 |
| ResNet-110 | Acc.(%) | 93.61 | 94.49 | 94.27 | 94.56 | 94.71 |

ure 1), we scale the 'K_n kernel' by the ratio of R which can be expressed as follows:

$$R = \frac{\text{Best_L1_norm}}{K_n_L1_norm}. \quad (2)$$

In addition, in the proposed method, the reactivation is performed per hyperparameter E epoch (i.e., at each E -th epoch) so that the reactivation can proceed according to the user environment. Using the proposed kernel reactivation, it is possible to reuse the invalid kernel for training, and prevent the collapse of the features extracted by the kernel by preserving the ratio of the weights of the invalid kernel. In addition, the proposed method has excellent adaptability, in that it only adds a negligible amount of computation and can be adaptively applied with the E cycle according to the network size and data size.

Experimental results: To evaluate the proposed method, we trained various ResNet models [12] with Cifar-10 [13] and Tiny-ImageNet [14] on a single RTX-2080 GPU. The Cifar-10 dataset contains 50,000 training images and 10,000 test images for 10 classes. Tiny-ImageNet is a subset of ImageNet and contains 100,000 training images and 10,000 validation images for 200 classes. The detailed environment settings for Cifar-10 training are as follows: The training is based on the stochastic gradient descent (SGD) optimizer with a batch size of 128, momentum of 0.9, weight decay of 0.0005, overall training epoch of 200, learning rate starting from 0.1, and cosine annealing scheduler is used. The environment settings for Tiny-ImageNet training are as follows: A batch size of 256 and 96 is used for ResNet18 and ResNet50, respectively, and a weight decay of 0.0001 and an overall training epoch of 90 are used. The rest of the settings are the same as the Cifar-10 environment setting. The baseline in the following experimental results table denotes the network trained from scratch without reactivation.

Tables 1–3 represent the results of evaluating ResNet-32 and ResNet-110 on the Cifar-10 dataset according to the four hyperparameters (i.e., α , β , γ , E) required in the proposed method. Table 1 shows the experimental result of β in C_2 (i.e., the influence of the filter in the layer). The remaining three hyperparameters, (α , γ , E), are set to (0.2, 0.05, 10), respectively. The experimental result indicates that the β derives optimal results at 0.3. If β is too small, such as 0.01, the performance improvement is not significant because of the little number of invalid kernels reactivated, and if it is set too high like 0.9, the filter that has an influence in the layer also becomes reactivated; hence, the performance improvement is less. Table 2 represents the experimental results related to the hyperparameter α and γ adopted in C_1 and C_3 , respectively, and the other two hyperparameters are evaluated with (β , E) = (0.3, 10). For ResNet-110, where the size of the network is relatively large, (α , γ) = (0.2, 0.1) shows the best performance, and for ResNet-32, which is a relatively small network, (α , γ) = (0.2, 0.05) has the highest accuracy. This indicates that Cifar-10 has more invalid kernels in ResNet-110 compared to ResNet-32, and it can be concluded that when small-size datasets such as Cifar-10 are evaluated on multiple networks, the number of invalid kernels increases as the network size increases. However, if the data size is large (i.e., Tiny-ImageNet), the number of invalid kernels may decrease; therefore, the hyperparameters should be adaptively reduced. Table 3 shows the experimental results related to E ,

Table 3. Accuracy of Cifar-10 according to E

| Network | Metric | Threshold E | | | | | |
|------------|---------|-------------|-------|-------|--------------|-------|-------|
| | | Baseline | 1 | 5 | 10 | 25 | 50 |
| ResNet-32 | Acc.(%) | 93.30 | 93.31 | 93.35 | 93.91 | 93.71 | 93.56 |
| ResNet-110 | Acc.(%) | 93.61 | 94.36 | 94.55 | 94.71 | 94.56 | 93.82 |

Table 4. Accuracy comparison on the Cifar-10 dataset

| Network | Metric | Method | | | |
|------------|---------|------------|------------|-------------------|-------------------|
| | | Baseline | WE [8] | FG [9] | Proposed |
| ResNet-32 | Acc.(%) | 93.30±0.08 | 93.50±0.03 | 93.83±0.07 | 93.91±0.06 |
| ResNet-110 | Acc.(%) | 93.61±0.15 | 94.00±0.07 | 94.73±0.11 | 94.71±0.10 |

Table 5. Accuracy comparison on the Tiny-ImageNet dataset

| Network | Metric | Method | | | |
|------------|---------------|------------|------------|------------|-------------------|
| | | Baseline | WE [8] | FG [9] | Proposed |
| ResNet-18 | Top-1 Acc.(%) | 60.31±0.07 | 60.72±0.12 | 61.12±0.15 | 61.32±0.09 |
| | Top-5 Acc.(%) | 81.81±0.14 | 82.10±0.13 | 82.52±0.16 | 82.53±0.13 |
| ResNet-50 | Top-1 Acc.(%) | 62.59±0.13 | 63.23±0.27 | 64.25±0.12 | 64.40±0.20 |
| | Top-5 Acc.(%) | 83.65±0.20 | 83.77±0.23 | 84.12±0.21 | 84.55±0.15 |
| ResNet-101 | Top-1 Acc.(%) | 64.90±0.59 | 65.71±0.58 | 66.66±0.41 | 66.72±0.40 |
| | Top-5 Acc.(%) | 85.07±0.33 | 85.39±0.46 | 85.76±0.33 | 85.97±0.38 |

which is a hyperparameter representing the cycle of reactivation, and the other three hyperparameters are set to $(\alpha, \beta, \gamma) = (0.2, 0.3, 0.1)$. In all cases, $E = 10$ achieved the highest performance. When $E = 1$, reactivation is performed too frequently; therefore, value changes continuously before the model is trained to the optimal value after reactivation. When $E = 50$, it is difficult to expect a significant performance improvement because of the less reactivation performed. Accordingly, hyperparameter settings are required according to the appropriate network size and data size. Empirically, hyperparameters settings $(\alpha, \beta, \gamma, E) = (0.2, 0.3, 0.1, 10)$ achieved the best performance; thus, we proceed with subsequent experiments with optimal hyperparameter settings.

Table 4 presents the comparison results of the proposed and existing methods on two networks, ResNet-32 and ResNet-110, with the Cifar-10 dataset. The hyperparameter setting is the same as that from Tables 1–3. To provide a more convincing analysis of our results, we conduct three experiments on each network and present the mean and standard deviation of accuracy in Table 4. In ResNet-32, the proposed method achieved the best performance of 93.91% with an accuracy increase of 0.61% compared to the baseline. And in ResNet-110, the accuracy increased by 1.1% compared to the baseline, and consequently, performance similar to that of FG [9], which uses twice the memory, is achieved. It should be noted that the increase in hardware resources, including memory usage required for training, causes environmental problems such as a significant amount of CO_2 emission as well as a cost burden [15].

Table 5 shows the comparison results for the mean and standard deviation of accuracy with Tiny-ImageNet, a larger dataset than Cifar-10, in ResNet-18, ResNet-50, and ResNet-101, respectively. Because the size of the dataset increases compared to Cifar-10, hyperparameter setting is performed as $(\alpha, \beta, \gamma, E) = (0.1, 0.3, 0.05, 10)$. The proposed method improves the top-1 accuracy and top-5 accuracy by 1.06% and 0.72% compared to the baseline in ResNet-18, and achieves a performance improvement of 1.87% and 1.03%, respectively, in ResNet50 compared to the baseline. In ResNet-101, the proposed method also shows significant accuracy enhancement compared to the baseline. Accordingly, compared to the existing methods, the proposed method indicates superior performance improvement with less memory use in all networks. It is noteworthy that the performance improvement effect of the proposed method becomes more excellent as the size of the dataset increases.

Conclusion: In this study, we proposed a kernel reactivation method to improve the performance of over-parameterized networks. The proposed method can be applied to the network adaptively after selecting an invalid kernel to perform reactivation through three conditions, considering the learning rate. Compared to the existing reactivation method, it exhibited the best performance in terms of trade-off of accuracy and hardware resource.

Acknowledgements: This work was partly supported by Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (No. 2020-0-01304, Development of Self-Learnable Mobile Recursive Neural Network Processor Technology) and the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education under Grant NRF-2019R1A6A1A03032119.

Data availability statement: The databases that support the findings of this study are openly available by their references.

© 2022 The Authors. *Electronics Letters* published by John Wiley & Sons Ltd on behalf of The Institution of Engineering and Technology.

This is an open access article under the terms of the Creative Commons Attribution License, which permits use, distribution and reproduction in any medium, provided the original work is properly cited.

Received: 25 April 2022 Accepted: 1 July 2022

doi: 10.1049/ell2.12578

References

- Choi, J., Chun, D., Kim, H., Lee, H.J.: Gaussian yolov3: An accurate and fast object detector using localization uncertainty for autonomous driving. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 502–511. IEEE, Piscataway (2019)
- He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask r-cnn. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 2961–2969. IEEE, Piscataway (2017)
- Liu, Z., Mao, H., Wu, C.Y., Feichtenhofer, C., Darrell, T., Xie, S.: A convnet for the 2020s. arXiv preprint, arXiv:220103545 (2022)
- Luo, J.H., Wu, J., Lin, W.: Thinet: A filter level pruning method for deep neural network compression. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 5058–5066. IEEE, Piscataway (2017)
- Han, S., Pool, J., Tran, J., Dally, W.: Learning both weights and connections for efficient neural network. In: Advances in Neural Information Processing Systems 28. MIT Press, Cambridge (2015)
- Li, H., Kadav, A., Durdanovic, I., Samet, H., Graf, H.P.: Pruning filters for efficient convnets. arXiv preprint, arXiv:160808710 (2016)
- Anwar, S., Hwang, K., Sung, W.: Structured pruning of deep convolutional neural networks. *ACM J. Emerg. Technol. Comput. Syst. (JETC)* **13**(3), 1–18 (2017)
- Lin, Z., Guo, K., Xing, X., Xu, X.: Weight evolution: Improving deep neural networks training through evolving inferior weight values. In: Proceedings of the 29th ACM International Conference on Multimedia, pp. 2176–2184. ACM, New York (2021)
- Meng, F., Cheng, H., Li, K., Xu, Z., Ji, R., Sun, X., et al.: Filter grafting for deep neural networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 6599–6607. IEEE, Piscataway (2020)
- Ye, J., Lu, X., Lin, Z., Wang, J.Z.: Rethinking the smaller-norm-less-informative assumption in channel pruning of convolution layers. In: Proceedings of the International Conference on Learning Representations. ICML, San Diego (2018)
- Loshchilov, I., Hutter, F.: Sgdr: Stochastic gradient descent with warm restarts. arXiv preprint, arXiv:160803983 (2016)
- He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778. IEEE, Piscataway (2016)
- Krizhevsky, A.: Learning multiple layers of features from tiny images. Master's Thesis, University of Tront (2009)
- Le, Y., Yang, X.: Tiny imagenet visual recognition challenge. *CS 231N* **7**(7), 3 (2015)
- Cai, H., Gan, C., Wang, T., Zhang, Z., Han, S.: Once-for-all: Train one network and specialize it for efficient deployment. arXiv preprint, arXiv:190809791 (2019)