Importance Resides In Activations: Fast Input-Based Nonlinearity Pruning

Baptiste Rossigneux, Vincent Lorrain, Inna Kucher, Emmanuel Casseau

Importance Resides In Activations: Fast Input-Based Nonlinearity Pruning

Baptiste Rossigneux[1], Vincent Lorrain[1], Inna Kucher[1], and Emmanuel Casseau[2]

[1] Université Paris-Saclay, CEA, LIST, F-91120
{baptiste.rossigneux,inna.kucher,vincent.lorrain}@cea.fr
[2] Université de Rennes, CNRS, INRIA
{emmanuel.casseau}@irisa.fr

Abstract. Deep learning has achieved tremendous successes across a broad range of applications, especially in computer vision with Convolutional Neural Networks (CNNs), which consist of successions of linear and nonlinear operations. In this study, our key contribution is a new procedure to linearize CNNs, in the most cost-effective way possible. We leverage information from the inputs to each nonlinear functions to identify which nonlinearities are less critical for the network's performance. Our method is versatile, adaptable to any common nonlinearity and CNN architecture. While it gives a small drop in accuracy across a wide range of CNNs with respect to state-of-the-art methods, it bypasses the usual significant computational effort to determine removable nonlinearities, whether layer-wise or channel-wise. Additionally, we provide a comprehensive analysis of network behavior during pruning, offering insights into internal damage, recovery, and effective retraining strategies.

Keywords: Deep learning · CNNs · Linearization · Pruning.

1 Introduction

Pruning forms a crucial part of the methods for reducing inference costs in deep learning models, alone, or combined with quantization and early exit strategies [28, 24]. Like most deep learning architectures, convolutional neural networks (CNNs) are composed of a succession of linear and nonlinear operations. While most pruning techniques have traditionally focused on the weights of the linear operations, our work explores the significant potential of nonlinearity pruning, especially for depth reduction. Depth reduction, consisting of pruning layer or groups of layers, is particularly relevant for minimizing floating-point operations (FLOPs), a common bottleneck in reducing latency, especially in embedded systems. Such strategies to network pruning and computation reduction of the nonlinear computations represent a promising development in pruning and computation reduction in neural networks.

Our key contribution is the Negative Positive Ratio (NPR), a metric to measure the importance of a nonlinearity layer, in the most cost-effective and transparent way possible. We show the effectiveness of our metric to pick the best set

of nonlinearities to prune, and explain our rationale behind the decisions taken for retraining. We test its performance on a diverse range of networks, from ResNet18 to more optimized and state-of-the-art CNNs like Efficient-Net or MobileOne, demonstrating the robustness of our approach across a wide range of CNNs. This is followed by a detailed discussion of the NPR metric and its comparative advantages. Subsequent sections will present our experimental setup, results, and a comprehensive analysis of the impact of our method on model performance.

2 Related works

Much of the interest in removing nonlinearity arises from private inference (PI), where a privatized model processes confidential data. Methods that allow private inference like homomorphic encryption suffer from an augmentation of the latency of nonlinear operations by one or two orders of magnitude. That is why the field of PI is much more interested in reducing the effective number of nonlinear operations produced than in other latency-critical fields such as edge AI : in traditional neural network inference, the nonlinearity is usually the cheapest operation. [19, 22, 5] use the sensitivity to pruning to attribute an importance score of a ReLU, based on empirical findings. The algorithm in [26] learns to replace nonlinearities by polynomials, to reach exceptionally low numbers of nonlinearity operations.

However, nonlinearity pruning can also be useful to optimize traditional neural network inference outside of PI, but only if more structured. Indeed, if an entire layer of nonlinearity is pruned, then the two layers surrounding it can be fused. Furthermore, depending on the layers, the resulting layer can take more memory but remain more efficient than the former linear - nonlinear - linear computation (for example fusing two 3x3 convolutional layers results in a 5x5 convolutional layer), or it can result in the simple pruning of one layer (for example a 3x3 convolutional layer can fuse with a 1x1 layer, resulting in only having the 3x3 convolution left). [12]. Instead of fusing, [23] directly insert the block they want, sum the activation of the usual block, and of the compressed block, while progressively increasing the influence of the latter. Other approaches from [31, 3, 9] transform the ReLU function by a Parametric-ReLU, with a learned slope guided towards 1. The goal is to let the network learn which ReLUs to transform to the identity function. Moreover, direct approaches on depth reduction have been actively explored [11, 8], with notable recent use of cosine distance between outputs as measures of layer importance by [13], demonstrating that depth is as critical as width for effective pruning. These structured pruning strategies highlight the efforts to bridge the gap between structured and unstructured pruning, the latter historically having been more accurate but less hardware-friendly. Our approach aligns with these structured pruning efforts.

Network linearization has been studied theoretically with the help of the Neural Tangent Kernel (NTK) [18], showing that nonlinearities only help during training [27], but can most certainly be pruned afterwards [1]. Although still a

work in progress, several work have introduced metrics to bridge the gap between nonlinearity measures and accuracy on a task : [1] measures the average path length, or the average number of times that data is processed by a nonlinearity, or counting the number of linear regions in the framework of Spline theory [2, 16, 17]. A recent approach by [4] successfully employs the Wasserstein metric from optimal transport theory to characterize nonlinearity by measuring the distance to the best affine fit. They however focus on analyzing the evolution of CNNs through the years rather than determining importance for later removal. While their metric is more general than ours presented below because it takes into account the shape of the nonlinearity, it is computationally intensive. Under the realistic assumption that a CNN maintains consistent nonlinearity types throughout its architecture, our metric focuses effectively on input/output distributions, with a more tractable solution, showing that it can be used for linearization.

Despite these theoretical advancements, the field is still dominated by learned methods for both layer pruning and private inference. While effective, these approaches often make the linearization process opaque and difficult to interpret. In contrast, our work aims to provide a transparent and interpretable method for nonlinearity pruning, addressing a key limitation in current approaches while maintaining efficiency and accuracy.

3 Methodology

In our study, we selected a diverse array of CNNs to validate the effectiveness and versatility of our nonlinearity pruning method. We chose ResNet18 [14] as the most used neural network to show a comparison with most other works available. We also picked ConvNext [25] to use a state-of-the-art CNN that adopts GeLU as nonlinearity to demonstrate the versatility of our method. MobileOne [30], built mostly like RepVGG [7] was used to showcase our technique on a state-of-the-art CNN that is perfectly suited for our method, since the network follows almost entirely the pattern : 3x3 convolution - ReLU - 1x1 convolution, allowing a perfect fusion if the ReLU is pruned. Finally we chose EfficientNet-lite[3] [29] to show that our techniques are competitive with more block-oriented techniques, such as [12]. We always suppose that we have a checkpoint ready for our desired dataset and do not consider the details of the original training.

3.1 NPR : A metric to measure nonlinearity importance

In developing an efficient method to identify non-essential nonlinearities within CNNs, we introduced the Negative Positive Ratio (NPR), a directly observable metric that does not rely on multiple epochs. This metric facilitates quick decisions about which nonlinear layers to prune, enhancing computational efficiency

[3] https://blog.tensorflow.org/2020/03/higher-accuracy-on-vision-models-with-efficientnet-lite.html

without extensive resource investment. At first, we based our intuition on the binary effect of a ReLU to be as cost effective as possible, by simply counting the zeros produced by a ReLU. This approach did not take into account the value of activations itself. A zero produced from an initial value value close to zero will have a much lesser impact on the activation distribution than from a value far from zero. Hence our metric becomes the absolute sum of the negative values. After observing the almost-gaussian distributions of the activations, we see that we are biased towards removing the layers with smaller gaussians, which does not account for the natural decrease in the number of values in a CNN. This leads to our consideration of 'stages' in the network. In CNNs, a stage typically consists of one or more layers grouped together, culminating in a significant feature extraction before potentially downsampling the spatial dimensions of the data. Each stage, therefore, acts as a functional unit, and should be considered as a different context from the others. That is why the first metric we come up with, NPR from equation 1, works to an extent, but tends to decimate the nonlinearities in entire stages, which helps realizing that this metric is stage-wise biased. The answer to this issue is equation 2, which gives the normalized NPR (NNPR) over the whole stage, by dividing a given NPR by the sum of the NPRs of its stage :

Let $A^- = \{i : i < 0\}$ and $A^+ = \{j : j > 0\}$.

$$NPR = \frac{\sum_{i \in A^-} \|i\|}{\sum_{j \in A^+} j} \tag{1}$$

$$NNPR = \frac{NPR}{NPR_{\text{stage}}}, \quad \text{with } NPR_{\text{stage}} = \sum_{i \in \text{stage}} NPR_i \tag{2}$$

with A^- and A^+ being respectively the sets of negative and positive elements of the inputs of a nonlinearity. Figure 1 illustrates the NPR and NNPR for a ResNet model trained on the CIFAR100 dataset. This figure underscores an additional advantage of our metric: it provides explanatory insights into the potential impact of pruning specific nonlinearities. By estimating the potential damage from removing particular nonlinearities, our approach aids in making informed decisions about which layers to prune. For example, the analysis indicates that removing any of the {3, 7, 11, 13}th ReLUs could significantly impair the network's performance. Therefore, our results on this example suggest that pruning could reasonably be limited to retaining only four specific layers without adversely affecting overall network effectiveness.

We need an approximation of the NNPR over the entire dataset, that only needs a fraction (typically 0.1%) of the entire dataset to converge to a convenient approximation. This is orders of magnitude faster than the several epochs needed by most state of the art techniques to evaluate the nonlinearities to prune, or to estimate their nonlinearity budget.

Most of the experiments in the paper focus on ReLU but the NNPR can generalize on other nonlinearities among the most common such as GeLU or SiLU, where the negative values are not set to zero but transformed by a smooth exponential-like function, since they approach zero for large negative inputs. Our

approach however would have to be adapted to activations that don't have this property, such as LeakyReLU or ELU [10].

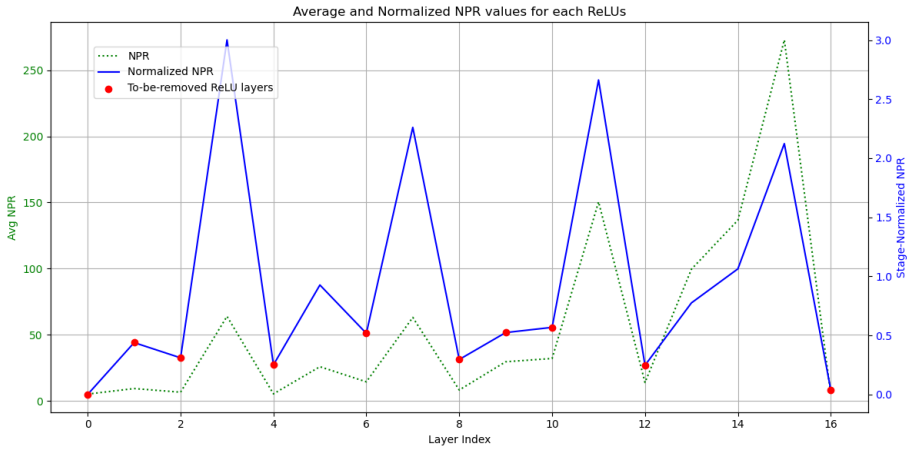

Fig. 1. The layer-wise value of average NPR and NNPR of each ReLUs of ResNet18, computed on 1000 samples of CIFAR100. Red dots are the ReLUs that will be pruned.

3.2 Training setup

For reasons explained at the end of this section, we trained after removing every ReLUs all at once. We found that the retraining can recover most of the accuracy in a limited number of epochs, but still can benefit for longer retrainings. As for the retrainings, we use the cosine scheduler and start with Lr=5e-4. We use Hinton knowledge distillation [15] with the logits of the original network since it improves systematically our results.

Inspired by the work of Kundu et al. [22], we incorporated a 3-way composite loss, which includes Hinton's knowledge distillation and PRAM loss, characterized by the sum of layer-wise normalized L2 distances post-nonlinearity activation (see Equation 3). It incentivizes the network to more closely align its activations with those of the teacher. We use $\lambda = 1$ and $\beta = 0.1$. Notably, the PRAM loss was omitted from the layer-wise pruning strategy due to its consistent negative impact on performance, suggesting it is effective only on very low nonlinearity budgets.

$$Loss = Loss_{CE} + \lambda Loss_{KD} + \beta \underbrace{\sum_{m \in I} \left\| \frac{\Psi^m_{original}}{\left\| \Psi^m_{original} \right\|_2} - \frac{\Psi^m_{pruned}}{\left\| \Psi^m_{original} \right\|_2} \right\|_2}_{PRAMLoss} \tag{3}$$

3.3 The case for progressive pruning

Guided by the intuition that the network would not be able to come back to its optimized, previously reached local minima due to its diminished expressivity, we tried to be as progressive as we could in damaging the network's accuracy. After having computed the set of nonlinearities to remove, we would prune them one by one, from the ones with the lower metric to the highest. We set a number of epochs between each nonlinearity pruning to have an equivalent number of epochs as usual, and use a cosine scheduler with a period equal to this number of epochs. One could object that the distribution of NNPRs could change during training, and the next lowest NNPR could change from one epoch to another. Empirically, we observed that the distribution does not change enough to change the order of progressive pruning. Contrary to our intuition, we observed a significant drop in accuracy compared to the same training budget. Figure 2 highlights this phenomenon between progressive runs (orange and blue) and a run when nonlinearities are pruned simultaneously (green).

4 Results

4.1 Results on layer-wise pruning

Table 1 shows the results of using NNPR to prune nonlinearities layer-wise on CIFAR100 [21] and the more challenging ImageNet-1k [6]. To compare fairly to a block-wise method [12] we count the number of equivalent pruned ReLU layers in their two extremes : EffLite3-DS-A (30 nonlinearities pruned) and EffLite3-DS-D (18 nonlinearities pruned). We apply the exact same training, but use our lighter method to find the nonlinearities to prune. We end up with slightly less accurate models, going from 77.8% to 77.5% (-0.3%) and from 76.8% to 76.2% (-0.6%), respectively. The clear asset of our technique here is not needing to do an expensive search over several epochs for the learnable nonlinearity mask to converge, but only a limited number of inferences to get the NNPR.

We also show the consistency of our results on a more restricted number of epochs but on ImageNet-1k, with a diversity of architectures. Results on ConvNext, an architecture using GeLU as nonlinearity, show that our method is not limited to ReLUs. We maintained the exact same regimes to show that it is consistent for different architectures. However, this uniformity highlighted a greater discrepancy between the pruned and baseline models for ConvNext, which originally underwent a more complex and compute-intensive training.

4.2 Results on channel-wise pruning

On the field of private inference, the priority is to prune as many nonlinearity operations as possible, to reduce their cost at inference. To align with state of the art methods and give another perspective on the pertinence of our metric, we applied our exact methodology but computed our metric on the channels of our nonlinearities, while still normalizing them stage-wise, to prune channel-wise

Model	#ReLU (k)	Top 1 accuracy
Dataset : CIFAR100 (180 epochs)		
ResNet18 (baseline)	557	76.0%
ResNet18 (9/17)	205 (37%)	77.0%
ResNet18 (7/17)	172 (31%)	76.0%
ResNet18 (5/17)	131 (23%)	74.4%
Dataset : ImageNet-1k (180 epochs)		
EffNet-lite3 (baseline)	10896	78.9%
EffNet-lite3 (30/48)	6280 (58%)	77.5%
EffNet-lite3 (18/48)	2868 (26%)	76.2%
Dataset : ImageNet-1k (100 epochs)		
ResNet18 (baseline)	2308	69.8%
ResNet18 (7/17)	727	70.1%
MobileOne (baseline)	3788	71.4%
MobileOne (22/42)	1856 (49%)	70.6%
MobileOne (12/42)	1103 (29%)	69.4%
ConvNext-T (baseline)	8580	81.9%
ConvNext-T (10/18)	5419 (63%)	78.0%
ConvNext-T (8/18)	4215 (49%)	77.2%

Table 1. Results of Top 1 accuracies for 100 epochs of training. In parentheses with the model names are the numbers of maintained nonlinearity layers.

Method	#ReLU (k)	Baseline Top 1	Top 1
DeepReduce [19]	28.7	78.0%	68.6%
SENet [22]	24.6	78.0%	70.6%
Ours	21	76.0%	66.0%
Ours	27	76.0%	67.5%

Table 2. Results of Top 1 accuracies for 100 epochs of training on CIFAR100, with channel-wise ReLU pruning.

the lower NNPR values. This allows us to be more fine-grained and prune more nonlinearity operations to reach levels reached by state of the art for comparison. Our approach, which requires no training epochs to compute the channel-wise pruning mask, contrasts with methods proposed by Kundu et al. [22] and Jha et al. [19], offering a more efficient alternative. Although our results are slightly inferior to state of the art approaches [23], it is important to note that our baseline model differs from theirs, which influences comparative outcomes. The distance to their results still suggests that our technique is better-suited for layer-wise than channel-wise pruning.

5 Discussion

5.1 The effect of nonlinearity pruning on internal representations

To evaluate the impact of nonlinearity pruning on the network's internal representations, we employed the Centered Kernel Alignment (CKA) [20], the most used metric to quantify similarity between representations. We use it as a layer-wise similarity metric between the original and the pruned model throughout training. Insights from Figure 2 reveal distinct effects based on the order of nonlinearity pruning:

- In scenarios where nonlinearity pruning progresses from shallow to deep layers (blue), the initial training epochs show significant damage at shallower layers, while deeper layers also exhibit notable impacts.
- Conversely, removing nonlinearities beginning with the deeper layers (orange) results in early significant damage to deeper representations, with minimal impact on shallower layers.

These observations indicate that the detrimental effects of nonlinearity pruning are localized, without compelling adaptation in shallower layers. This localization explains why starting pruning from deeper layers yields a representation more similar to the original by the end of training, achieving higher accuracy. The progressive pruning appears to allow more gradual adaptation across the network.

Considering the ReLU functions as a noise canceller that zeros out negative signals, its pruning could be seen as reintroducing this 'noise' into the network's output. Hence, our metric effectively becomes an inverse signal-to-noise ratio (SNR) at each layer. A low SNR correlates with a higher value in our metric, suggesting a nonlinearity that is expendable. We tested this hypothesis by simulating nonlinearity pruning through the injection of noise equivalent to the negative part of a Gaussian distribution, observing similar dynamics but reduced performance, as this noise does not correlate with the original input signals.

5.2 Could we have trained a damaged CNN from scratch ?

If previous sections showed that we should not aim for lowest damage when pruning nonlinearities, it would make sense to wonder if this damage could simply be made at initialization, and if there is really any benefit at taking a pretrained network rather than training from scratch. As demonstrated in Table 3, there is in fact a substantial difference in final accuracy between starting from an optimized checkpoint compared to starting anew. This table further elucidates this by presenting an ablation study, highlighting the unique training decisions enabled by the availability of a pre-trained checkpoint. First, it shows that not using distillation because of a lack of a valuable checkpoint also hurts the most the performance. Another issue with nonlinearity pruning at initialization is that the NNPR distribution is far from its converged form, which leads to sub-optimal

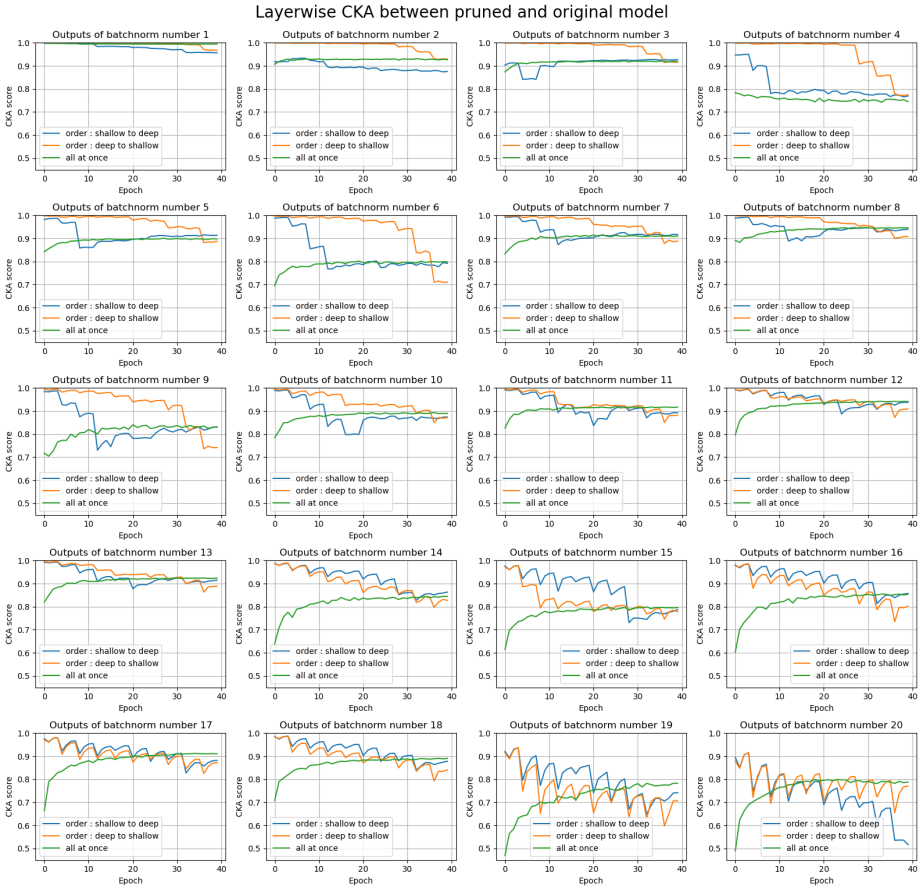

Fig. 2. The layer-wise CKA between representations of the original ResNet18 model on CIFAR100 and the version with pruned ReLUs, by 3 different methods : pruning ReLUs all at once (green), pruning ReLUs from the shallowest to the deepest (blue), and pruning from the deepest to the shallowest (orange), chosen according to their NNPR.

Model	Optimal NNPR	From a checkpoint	Distillation	Top 1 acc.
ResNet18 (7/17)	✓	✓	✓	76.0%
ResNet18 (7/17)	✓	✓	✗	71.4%
ResNet18 (7/17)	✓	✗	✓	72.7%
ResNet18 (7/17)	✗	✗	✓	72.3%
ResNet18 (7/17)	✗	✗	✗	69.9%

Table 3. Ablation study of multiple components of the nonlinearity pruning and retraining. Training is done on CIFAR100 for 180 epochs. "Optimal NNPR" designates whether the model pruned nonlinearities based on NNPR computed on a pretrained model, or on the untrained model.

choices for removing nonlinearities. As Figure 3 demonstrates, the NNPR distribution requires a significant number of epochs to stabilize—at least 50 epochs. This significant compute would be necessary for only modest gains in final accuracy, compared to pruning based on the NNPR values obtained at initialization, as highlighted by rows 3 and 4 of Table 3. The last asset is more trivially to start from a pretrained checkpoint. As hinted by [1], we further demonstrate that a first significant part of the training has to be conducted with the nonlinearities still in place. Together, these factors ensure our model retains high accuracy post-pruning, which would not be achievable if started from scratch.

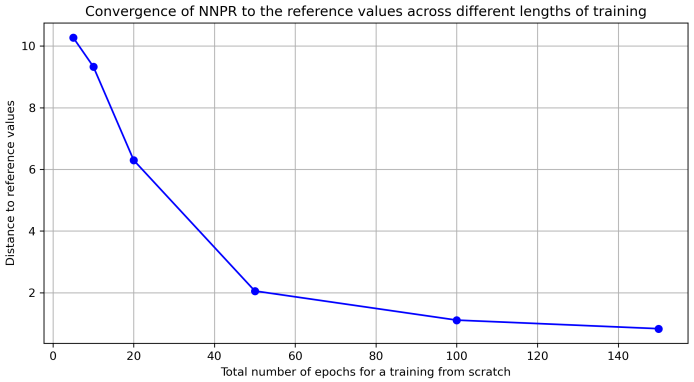

Fig. 3. L1 distance between the NNPR from our best ResNet18 checkpoint on CIFAR100 and NNPRs obtained for different lengths of training. We launched different trainings and varied the number of epochs to not be biased by factors such as the learning rate scheduler.

6 Conclusion and future works

We present a novel method for layer-wise or channel-wise pruning of nonlinearities, showing that simply observing the activations is enough to spot the less useful nonlinearities. We also explain our choices for retraining, by measuring the impact of each by observing the noise injection-like effects of nonlinearity pruning on internal representations. This technique allows for a faster and more transparent nonlinearity pruning for depth reduction by fusion of linear layers or private inference. Still, a better adaptation of our metric to channel-wise pruning is still to find, to close the gap with other methods. Further works will aim to go even more refined by simply looking at the network architecture, and the training algorithm, since the gap between evaluating the NNPR with samples from the training set or with random noise is not very substantial. Exploring this would help to design a double-purpose network : fit for training, but also already ready for depth compression.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Ali Mehmeti-Göpel, C.H., Disselhoff, J.: Nonlinear advantage: trained networks might not be as complex as you think. In: Proceedings of the 40th International Conference on Machine Learning. ICML'23, JMLR.org (2023)
2. Balestriero, R., et al.: A spline theory of deep learning. In: International Conference on Machine Learning. pp. 374–383. PMLR (2018)
3. Bhardwaj, K., Ward, J., Tung, C., Gope, D., Meng, L., Fedorov, I., Chalfin, A., Whatmough, P., Loh, D.: Restructurable activation networks (2022)
4. Bouniot, Q., Redko, I., Mallasto, A., Laclau, C., Struckmeier, O., Arndt, K., Heinonen, M., Kyrki, V., Kaski, S.: From alexnet to transformers: Measuring the non-linearity of deep neural networks with affine optimal transport. In: ICML 2024 Workshop on Mechanistic Interpretability
5. Cho, M., Joshi, A., Reagen, B., Garg, S., Hegde, C.: Selective network linearization for efficient private inference. In: International Conference on Machine Learning. pp. 3947–3961. PMLR (2022)
6. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition. pp. 248–255 (2009). https://doi.org/10.1109/CVPR.2009.5206848
7. Ding, X., Zhang, X., Ma, N., Han, J., Ding, G., Sun, J.: Repvgg: Making vgg-style convnets great again. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 13733–13742 (2021)
8. Dong, X., Chen, S., Pan, S.: Learning to prune deep neural networks via layer-wise optimal brain surgeon. Advances in neural information processing systems **30** (2017)
9. Dror, A.B., Zehngut, N., Raviv, A., Artyomov, E., Vitek, R., Jevnisek, R.J.: Layer folding: Neural network depth reduction using activation linearization. In: British Machine Vision Conference (2021), https://api.semanticscholar.org/CorpusID:235458222

10. Dubey, S.R., Singh, S.K., Chaudhuri, B.B.: Activation functions in deep learning: A comprehensive survey and benchmark. Neurocomputing **503**, 92–108 (2021), https://api.semanticscholar.org/CorpusID:250089226
11. Elkerdawy, S., Elhoushi, M., Singh, A., Zhang, H., Ray, N.: To filter prune, or to layer prune, that is the question. In: Asian Conference on Computer Vision (2020), https://api.semanticscholar.org/CorpusID:220496508
12. Fu, Y., Yang, H., Yuan, J., Li, M., Wan, C., Krishnamoorthi, R., Chandra, V., Lin, Y.: DepthShrinker: A new compression paradigm towards boosting real-hardware efficiency of compact neural networks. In: Chaudhuri, K., Jegelka, S., Song, L., Szepesvari, C., Niu, G., Sabato, S. (eds.) Proceedings of the 39th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 162, pp. 6849–6862. PMLR (17–23 Jul 2022), https://proceedings.mlr.press/v162/fu22c.html
13. Gromov, A., Tirumala, K., Shapourian, H., Glorioso, P., Roberts, D.A.: The unreasonable ineffectiveness of the deeper layers (2024)
14. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) pp. 770–778 (2015), https://api.semanticscholar.org/CorpusID:206594692
15. Hinton, G., Vinyals, O., Dean, J.: Distilling the knowledge in a neural network (2015)
16. Humayun, A.I., Balestriero, R., Balakrishnan, G., Baraniuk, R.: Splinecam: Exact visualization and characterization of deep network geometry and decision boundaries. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (2023)
17. Humayun, A.I., Balestriero, R., Baraniuk, R.G.: Training dynamics of deep network linear regions. ArXiv **abs/2310.12977** (2023), https://api.semanticscholar.org/CorpusID:264305887
18. Jacot, A., Gabriel, F., Hongler, C.: Neural tangent kernel: Convergence and generalization in neural networks. Advances in neural information processing systems **31** (2018)
19. Jha, N.K., Ghodsi, Z., Garg, S., Reagen, B.: Deepreduce: Relu reduction for fast private inference. In: International Conference on Machine Learning. pp. 4839–4849. PMLR (2021)
20. Kornblith, S., Norouzi, M., Lee, H., Hinton, G.: Similarity of neural network representations revisited. In: International conference on machine learning. pp. 3519–3529. PMLR (2019)
21. Krizhevsky, A., Hinton, G., et al.: Learning multiple layers of features from tiny images (2009)
22. Kundu, S., Lu, S., Zhang, Y., Liu, J.T., Beerel, P.A.: Learning to linearize deep neural networks for secure and efficient private inference. In: The Eleventh International Conference on Learning Representations (2022)
23. Kundu, S., Zhang, Y., Chen, D., Beerel, P.A.: Making models shallow again: Jointly learning to reduce non-linearity and depth for latency-efficient private inference. 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW) pp. 4685–4689 (2023), https://api.semanticscholar.org/CorpusID:258331526
24. Liang, T., Glossner, J., Wang, L., Shi, S., Zhang, X.: Pruning and quantization for deep neural network acceleration: A survey. Neurocomputing **461**, 370–403 (2021)
25. Liu, Z., Mao, H., Wu, C.Y., Feichtenhofer, C., Darrell, T., Xie, S.: A convnet for the 2020s. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2022)

26. Peng, H., Huang, S., Zhou, T., Luo, Y., Wang, C., Wang, Z., Zhao, J., Xie, X., Li, A., Geng, T., Mahmood, K., Wen, W., Xu, X., Ding, C.: Autorep: Automatic relu replacement for fast private network inference. 2023 IEEE/CVF International Conference on Computer Vision (ICCV) pp. 5155–5165 (2023), https://api.semanticscholar.org/CorpusID:261049233
27. Roberts, D.A., Yaida, S., Hanin, B.: The principles of deep learning theory. ArXiv **abs/2106.10165** (2021), https://api.semanticscholar.org/CorpusID:235485320
28. Shen, Y., Sun, M., Zhao, J., Zou, A.: Chain of compression: A systematic approach to combinationally compress convolutional neural networks. ArXiv **abs/2403.17447** (2024), https://api.semanticscholar.org/CorpusID:268692052
29. Tan, M., Le, Q.: EfficientNet: Rethinking model scaling for convolutional neural networks. In: Chaudhuri, K., Salakhutdinov, R. (eds.) Proceedings of the 36th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 97, pp. 6105–6114. PMLR (09–15 Jun 2019), https://proceedings.mlr.press/v97/tan19a.html
30. Vasu, P.K.A., Gabriel, J., Zhu, J., Tuzel, O., Ranjan, A.: Mobileone: An improved one millisecond mobile backbone. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 7907–7917 (2023)
31. Wu, J., Zhu, D., Fang, L., Deng, Y., Zhong, Z.: Efficient layer compression without pruning. Trans. Img. Proc. **32**, 4689–4700 (jan 2023). https://doi.org/10.1109/TIP.2023.3302519, https://doi.org/10.1109/TIP.2023.3302519