Proceedings of the 39th Chinese Control Conference
July 27-29, 2020, Shenyang, China

Convolutional Neural Network Pruning: A Survey

Sheng Xu[1], Anran Huang[1] Lei Chen*[1,2] Baochang Zhang[1]
1. School of Automatic Science and Eletrical Engineering, Beihang University, Beijing, 100191, China
E-mail: {shengxu@buaa.edu.cn, huanganran@buaa.edu.cn, bczhang@buaa.edu.cn}
2. State Key Laboratory of Software Development Environment
Beijing Advanced Innovation Center for Big Data and Brain Computing, Beijing, 100191, China
E-mail: Alexander_Lei.CHEN@outlook.com

Abstract: Deep convolutional neural networks have enabled remarkable progress over the last years on a variety of visual tasks, such as image recognition, speech recognition, and machine translation. These tasks contribute many to machine intelligence. However, developments of deep convolutional neural networks to a machine terminal remains challenging due to massive number of parameters and float operations that a typical model contains. Therefore, there is growing interest in *convolutional neural network pruning*. Existing work in this field of research can be categorized according to three dimensions: pruning method, training strategy, estimation criterion.

Key Words: convolutional neural networks, machine intelligence, pruning method, training strategy, estimation criterion

1 Introduction

With the development of convolutional neural networks (CNNs), computer vision tasks have achieved unprecedented levels of accuracy. CNNs have advanced image classification[14, 18, 38, 39], semantic segmentation[8, 28, 30] and object detection[5, 26, 35, 36] to level which is unachievable with other methods. The success of CNNs in perceptual tasks is largely due to its huge amount of parameters and computation. To improve performance, deeper and wider networks have been designed, which increases the demand for computational power. However, the drive to improve accuracy often comes at a cost: modern networks require high computational resources beyond the capabilities of many practical applications. To overcome this problem, convolutional neural network pruning shows it extraordinary capability to reduce the CNNs' computation, which makes CNNs feasible to be implemented on mobile and embedded devices.

Already by now, convolutional neural network pruning has achieved outstanding performance on different tasks, such as image classification[17, 21, 23] and object detection[25]. We categorize methods for convolutional neural network pruning according to three dimensions as follows:

- **Pruning Method:** The pruning method decides the training strategy and estimation criterion in advance. For different pruning method, the pruned model can be non-structured deformed or structured deformed. Hence, pruning method can be categorized into non-structured pruning and structured pruning.
- **Training Strategy:** The training process is employed to make the parameters feasible to prune, whose strategies can be categorized into *hard*, *soft* and redundant approaches.
- **Estimation Criterion:** The estimation criterion can be designed via multiple algorithms. They can be set in forward and backward propagation, making up the train process accompanied by training strategy.

We refer to Figure 1 for an illustration. The survey is also structured according to these three dimensions: we start with discussing pruning methods in Section 2, cover training strategies in Section 3, and outline estimation criterions in Section 4. We conclude with an outlook on future directions in Section 5.

2 Pruning Method

Pruning method defines the process of the trained parameter set. More specifically, which and what kind parameter to prune. It also influences many on the pruned structure. We discuss the pruning methods, categorize them to two main classes.

2.1 Non-Structured Pruning

Early work of CNN pruning research concentrates on the weights of convolution, for the number of model parameters is not so enormous. For example, LeNet [19] uses convolution and pooling to extract features. Because the input is single channel image, the structure takes 2.19M float operations (FLOPs) only. Weight pruning zeroizes the unnecessary connections accounting for a large proportion of required computation during execution as shown in Figure 2 (b). For the architecture consistency, weight can only be zeroized rather than pruned. Hence, weight pruning needs special coordinate for every weight, which is difficult to satisfy in nowaday trillion-level model. Moreover, special hardware is also required to accelerate the training process. Early work in weight pruning [20] proposes a saliency measurement to remove redundant weights determined by the second-order derivative matrix of the loss function. Numerous methods have been proposed to determine the weight zeroizing criterion, such as iterative thresholding selection [13] and Hoffman code [12]. Incorrect weight pruning is the main problem in weight zeroizing. To solve the problem, Guo *et al.* [11] propose a connection splicing method, which is proved to be effective.

Kernel is a structure of CNNs, but is unable to be removed after being zeroized in kernel-based pruning. The structural consistency of CNN will be satisfied only if kernels of same input feature map are trimmed simultaneously, which is difficult to satisfy. Hence, the zero matrix is used to denote the pruned kernel for structural consistency as shown in Figure

* Contact Author.

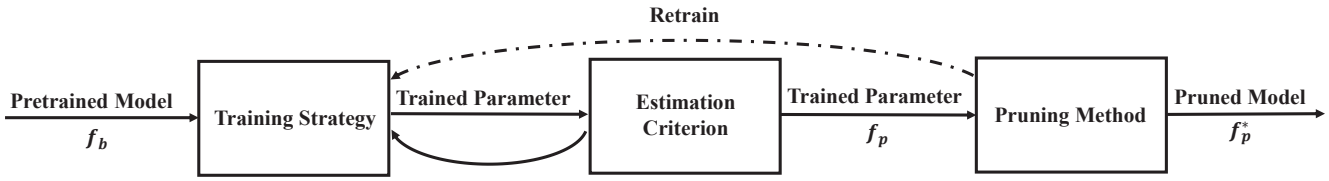

Fig. 1: Abstract illustration of Convolutional Neural Network Pruning. Input is a pretrained model f_b. The training strategy and estimation criterion are used to prduce a trained parameter set f_p. Then pruning method is to prune the network according to f_p. Finally, the output is pruned model f_p^*.

2 (c), similar to non-structured weight pruning. Li *et al.* [21] propose a method based on kernel sparsity which classify the kernels via sparsity and produce a sparser model. Kernel pruning greatly accelerate the non-structured pruning process, and can achieve the best trade-off between accuracy and pruning rate.

2.2 Structured Pruning

Structured pruning directly removes structured CNN parts to simultaneously compress and speedup CNNs and is well supported by various off-the-shelf deep learning libraries. Among the categories, filter pruning has attracted the most attention, which prunes filters integrally as shown in Figure 2 (d). As shown in Figure 3, single filter pruning compresses the output feature map dimensions. Moreover, the corresponding kernel in next layer should be pruned to keep the consistency of CNN architecture. Hence, the structured filter pruning is structured filter-kernel pruning. Structured filter-kernel pruning can be implemented not only after training but also during training, which will be further discussed in Section 3. Some methods prune filters from CNNs that are identified as having less information [15, 16, 21, 44]. The others prune filters from CNNs that are identified as having a small effect on the output accuracy, thus compressing the networks [17, 29]. The detailed classification will be discussed Section 4. Filter-kernel pruning is widely implemented for it can convey a new architecture and is feasible to all CNN models. Moreover, it can efficiently prune the width redundancy of networks, resulting the large computation.

Block pruning is first proposed by Lin *et al.* [25]. Unlike other methods, structured block pruning is more like a remodeling process. The target is to prune the block wholly. To avoid the zero input, only the block with residual connection can be pruned. Despite its limitations, block pruning can effectively remove the depth redundancy for some special architectures. In addition, it can be implemented united with filter-kernel pruning, thus realizing a higher pruning rate.

As listed in Table 1, non-structured kernel pruning can achieve the best trade-off between accuracy and pruning rate. Among structured pruning, filter-kernel pruning achieves the better performance than block pruning. However, block pruning can realize the best training speed. To conclude, the better performance is achieved with a smaller pruning unit, which contrast the training speed.

Table 1: Results of different state-of-the-art pruning methods using ResNet-50 on ImageNet ILSVRC2012. P.R. denotes pruning rate in short.

Method	Base/Pruned Top1%	Base/Pruned Top5%	FLOPs P.R.%
Non-Structured Kernel Pruning			
KSE[22]	76.15/75.51	92.87/92.69	73.4
Structured Filter-Kernel Pruning			
C-SGD[3]	75.33/75.27	92.56/92.46	36.8
FPGM[16]	76.15/75.59	92.87/92.63	42.2
Structured Block Pruning			
GAL[25]	76.15/71.95	92.87/90.82	55.0

In next section we discuss training strategies that are decided by these kinds of pruning methods.

3 Training Strategy

We categorize the training strategies into three classes.

3.1 *Hard* Pruning Strategy

Hard pruning is mostly employed in prior works. The pruning process is implemented sequently with training process in *hard* pruning, id est, 'Retrain' procedure plotted in Figure 1. As shown in Figure 3, the first output feature map is blank for the first filter in l^{th} layer is pruned, and then the first kernel in filters $(l+1)^{th}$ layer will be pruned. Next, the network will be reconstructed by left layers and fine-tuned to recover its accuracy. Layer by layer, the training and pruning are finished simultaneously. Hence, *hard* pruning strategy can only be employed in structured pruning. Li *et al.* [21] first propose the method for *hard* pruning network. Luo *et al.* [29] use the *hard* pruning strategy to prune network and reconstruct the output simultaneously. Using the reconstruction error for optimal target, it gets the best accuracy among *hard* pruning.

However, *hard* pruning is not able to achieve satisfying performance as well as greatly time-consuming, for the repeated fine-tuning process.

3.2 *Soft* Pruning Strategy

In non-structured weight pruning, the element zero is employed to substitute the unnecessary weights, thus denote pruning the weights [12, 20]. Inspired by this, the *soft* mask is employed after convolutional layer to train the model for

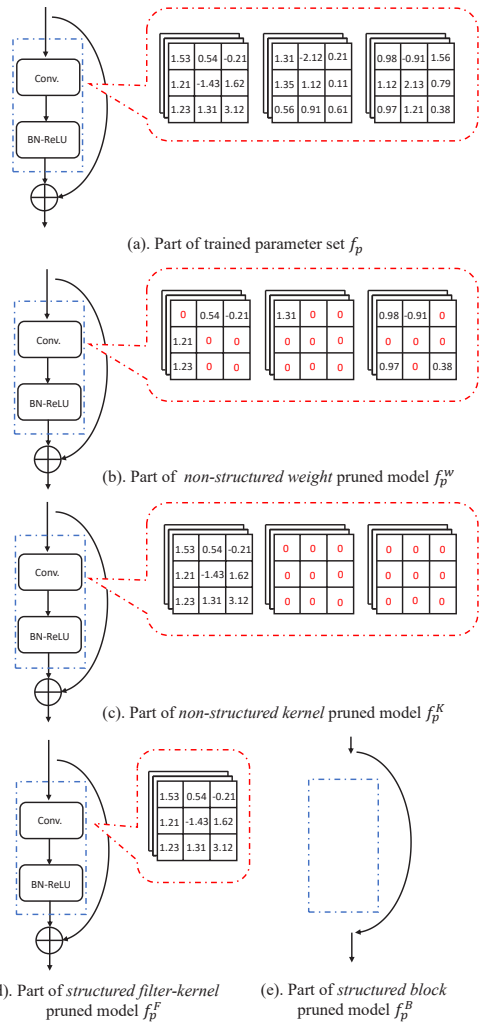

Fig. 2: Detailed categorization of pruning method. Take a single-layered residual block in ResNet for example. Conv. denotes convolution operation, BN-ReLU denotes the batch normalization and ReLU activation in sequence. The sub-figure (a) denotes a residual block after training. (b) and (c) denote the non-structured weight and kernel pruned model, respectively. (d) and (e) denote the structured filter-kernel and block pruned model, respectively.

filter-kernel pruning as plotted in Figure 4 (a) and (b). Categorized by the element property of mask vector, *soft* mask can be binary and full-precision.

Binary mask consists of vector containing only 1 and 0 elements, which can be reasoned as an indicator. Binary mask refers to a optimization problem, [24] solve this by Taylor expansion efficiently. NISP [44] use a binary mask and importance backward propagation to search the important filter. SFP [15] use the ℓ_2 norm to evaluate the information of filters and binary mask to zeroize the output of less important ones. Similarly, FPGM [16] use the ℓ_2 norm geometric median for evaluation.

Full-precision mask is universally employed in *soft* pruning. The back propagation of full-precision mask can be identical as other parameters, however, normal SGD is proved defective [2]. It is a vector to scale the output. As shown in Figure 4 (b), the vector mask shape is used for

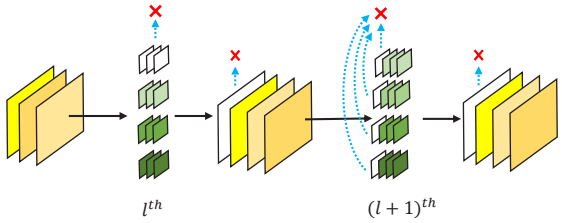

Fig. 3: Detailed structured filter-kernel pruning. Take two convolutional layers for example. The input-output dimension change from $3{-}4{-}4$ to $3{-}3{-}3$ as filter dimension change from $3{\times}4{-}4{\times}4$ to $3{\times}3{-}3{\times}4$.

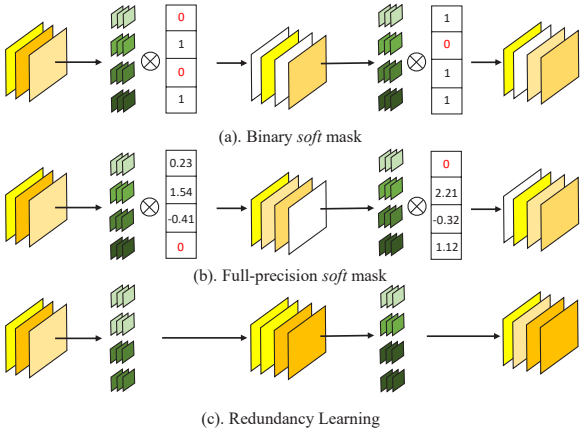

Fig. 4: Detailed training strategy. Take structured filter-kernel pruning for example. (a), (b) and (c) are binary *soft* mask, full-precision *soft* mask and redundant learning, respectively. Blank square denotes the blank feature map after scaling by zero.

filter-kernel pruning. Similarly, a cube and element mask can be employed for kernel and block pruning, respectively. He *et al.* [17] proposed a mask-based method, and use two step *LASSO* regression [40] to solve the optimal problem of mask. GAL [25] uses mask in both filter-kernel and block pruning, and employ *FISTA* [1, 9] algorithm to solve the optimal problem of mask.

3.3 Redundant Learning Strategy

Prior work is mostly based on the zeroizing out the redundant parameters. In contrast, a new novel theory on account of redundant learning is proposed, aiming to make the similar filters to the same and trims the redundant ones as plotted in Figure 4 (c). C-SGD [3] employs the redundant learning in back propagation. It clusters the filters via K-Means, and uses a special back propagation algorithm to make the filters in one cluster identical, thus trimming the redundant ones to compress the network.

As listed in Table 2, *soft* pruning and redundant learning can achieve the best performances in contrast to *hard* pruning. Also, *hard* pruning takes more training time than the other two. In next section, we will discuss about the estimation criterion for parameters, which influence the performance deeply.

Table 2: Results of different state-of-the-art training strategies using ResNet-50 on ImageNet ILSVRC2012. For the fair comparison, we choose the structured filter-kernel pruning and similar pruning rate. P.R. denotes pruning rate in short.

Method	Base/Pruned Top1%	Base/Pruned Top5%	FLOPs P.R.%
Hard Pruning			
ThiNet[29]	-/72.04	-/91.14	36.8
Soft Pruning			
CP[17]	-/72.30	-/90.80	33.2
FPGM[16]	76.15/75.59	92.87/92.63	42.2
Redundant Learning			
C-SGD[3]	75.33/75.27	92.56/92.46	36.8

Table 3: Results of different state-of-the-art estimation criterion using ResNet-50 on ImageNet ILSVRC2012. For the fair comparison, we choose the structured filter-kernel pruning and similar pruning rate. P.R. denotes pruning rate in short.

Method	Base/Pruned Top1%	Base/Pruned Top5%	FLOPs P.R.%
Importance-based Criterion			
FPGM[16]	76.15/75.59	92.87/92.63	42.2
Sparsity-based Criterion			
C-SGD[3]	75.33/75.27	92.56/92.46	36.8
Reconstruction-based Criterion			
GDFP[24]	76.15/72.61	92.87/91.05	39.1

4 Estimation Criterion

The train strategy discussed above aims to find a train method. To guide the train process, estimation criterion is employed. We category the estimation criterion into three classes: importance-based, reconstruction-based and sparsity-based.

4.1 Importance-based Criterion

Redundant parameters in pruning can be elements, matrices and tensors, which definitely have the norm information. Prior work pays most attention on the norm information. The universal criterion is that the larger norm comes with the more information. Hence, they tend to trim the parameters with smaller norm values. ℓ_1 norm can be employed in both non-structured weight pruning [13] and non-structured filter-kernel pruning [21], which subtly proves the universality of norm-based criterion. To further the success of importance-based criterion, p norm is widely employed. GDP [24] has discussed about the effectiveness of different p norm and explored the best capability of ℓ_2 norm. ℓ_2 norm is well received for its good derivability. He *et al.* [15] sort the filters via ℓ_2 norm value. However, is the norm value positive with the importance? FPGM [16] rebuts the ordinary thoughts and sorts filters by the ℓ_2 norm of filter-geometric median distance rather than absolute value. Other than norm based importance criterions, Yu *et al.* [44] introduce a importance back propagation method, which directly evaluate the importance via back propagation weights. Importance-based criterion is widely implemented for achieving fast computation. However, it compresses the high-dimensional information, which may account for some accuracy drop in the methods based on this criterion.

4.2 Sparsity-based Criterion

Sparsity-based criterion focuses on the higher-dimensional information. It aims to prune filters sharing similar information, which leads to a sparser architecture with less parameter and stronger feature extraction power. Li *et al.* [22] propose a non-structured kernel pruning manner based on kernel sparsity in forward propagation, which sparsifies the kernels of same input feature map and leads to a sparse pruned model. Ding *et al.* [3] propose a structured filter-kernel pruning manner based on high-dimensional position sparsity of filters in backward propagation, which sparsifies the filters in one layer. Sparsity-based criterion shows its potentiality on high-dimensional information interpretation. This kind of methods has achieved the lowest performance drop in both non-structured and structured pruning.

4.3 Reconstruction-based Criterion

In contrast to the two kinds mentioned above, reconstruction-based criterion focuses on the output feature map directly. The main idea is to minimize the reconstruction error. Thus the pruned model can be the optimal approximation to pretrained model. Luo *et al.* [29] propose the method using greedy search to find the filters less important to output. He *et al.* [17] employ the two-step *LASSO* regression to solve the reconstruction problem. Lin *et al.* [24] employ the *Taylor* expansion to solve the optimal problem. GAL [25] employs the *FISTA* [1, 9] algorithm to solve minimization of the reconstruction error. Reconstruction-based criterion involves the optimal problem of the sparse criterion, id est, ℓ_1 norm. Multiple algorithms are employed to solve the optimal problem such as *LASSO* and *FISTA*.

As listed in Table 3, importance-based and sparsity-based criterion can achieve the better performance than reconstruction-based criterion.

5 Future Directions and Conclusion

In this section, we discuss several current and future directions for research on CNN pruning. Most existing pruning methods are designed for image classification, which is usually regarded as a relatively small computational task comparing to other complex tasks. On the one hand, this provides satisfying performance as well as challenging benchmarks for pruning task based on image classification. On the other hand, it is relatively easy to find a well-suited pruning algorithm to begin the research on other tasks. We thus consider the applicability to implement pruning on other tasks. Notable first step in this direction has been taken in object detection in [25]. In comparison with other compression methods such as network quantization [10, 34, 42], pruning via GAL [25] can achieve the lower mAP drop. As listed in Table 4, pruning via GAL has achieve 2.5× acceleration, which is far from practical in embedded and mobile devices. In comparison , quantization method can achieve higher acceleration rate. Hence, for pruning methods, potentiality is still far from throughly released when implemented on de-

Table 4: Results of GAL pruning and network quantization methods using Faster R-CNN on VOC 2007test. Note that the pruning baseline is ResNet-50, quantization baseline is ResNet-34.

Method	Base/Compressed mAP%	mAP Drop%	Speedup
CNN Pruning			
GAL[16]	70.4/68.7	1.7	2.5×
CNN Quantization			
XNOR[34]	75.6/54.7	18.5	40×
TBN[41]	75.6/59.0	16.6	40×

tection tasks. For another potential application, 3D object categorization[6, 7] may be a compatible direction. Architectures [32, 33, 43] use for 3D object categorization mainly consist of multilayer perceptron (MLP) [37], whose computation is mainly resulted from fully-connection. Hence, non-structured weight pruning can be employed in 3D network pruning.

An additional promising direction is to develop CNN pruning methods together with the neural architecture search (NAS) [4, 27, 31, 45]. According to NAS, search space is of vital importance. If a manually designed search space is employed, redundancy will exist in the searched model. To solve this, CNN pruning is able to trim the redundant search space. In this way, the NAS architecture can reach a great trade-off between accuracy and computation.

We summarize recent efforts on pruning convolutional neural networks (CNNs) in three dimensions: pruning method, training strategy and estimation criterion. CNN pruning has achieved great success among various CNN compression approaches. For future research, we propose our advice on two promising directions, which will definitely further the success of CNN pruning.

References

[1] Amir Beck and Marc Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM journal on imaging sciences*, 2(1):183–202, 2009.

[2] Léon Bottou. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*, pages 177–186. Springer, 2010.

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

[4] Thomas Elsken, Jan Hendrik Metzen, and Frank Hutter. Neural architecture search: A survey. *Journal of Machine Learning Research*, 20(55):1–21, 2019.

[5] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, 88(2):303–338, 2010.

[6] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research*, 32(11):1231–1237, 2013.

[7] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3354–3361. IEEE, 2012.

[8] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 580–587, 2014.

[9] Tom Goldstein, Christoph Studer, and Richard Baraniuk. A field guide to forward-backward splitting with a fasta implementation. *arXiv preprint arXiv:1411.3406*, 2014.

[10] Jiaxin Gu, Ce Li, Baochang Zhang, Jungong Han, Xianbin Cao, Jianzhuang Liu, and David Doermann. Projection convolutional neural networks for 1-bit cnns via discrete back propagation. In *AAAI*, volume 33, pages 8344–8351, 2019.

[11] Yiwen Guo, Anbang Yao, and Yurong Chen. Dynamic network surgery for efficient dnns. In *Advances In Neural Information Processing Systems*, pages 1379–1387, 2016.

[12] 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*, 2016.

[13] Song Han, Jeff Pool, John Tran, and William Dally. Learning both weights and connections for efficient neural network. In *Advances in neural information processing systems*, pages 1135–1143, 2015.

[14] 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.

[15] Yang He, Guoliang Kang, Xuanyi Dong, Yanwei Fu, and Yi Yang. Soft filter pruning for accelerating deep convolutional neural networks. In *International Joint Conference on Artificial Intelligence*, 2018.

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

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

[18] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[19] Yann LeCun, Léon Bottou, Yoshua Bengio, Patrick Haffner, et al. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[20] Yann LeCun, John S Denker, and Sara A Solla. Optimal brain damage. In *Advances in neural information processing systems*, pages 598–605, 1990.

[21] Hao Li, Asim Kadav, Igor Durdanovic, Hanan Samet, and Hans Peter Graf. Pruning filters for efficient convnets. In *International Conference on Learning Representations*, 2016.

[22] Yuchao Li, Shaohui Lin, Baochang Zhang, Jianzhuang Liu, David Doermann, Yongjian Wu, Feiyue Huang, and Rongrong Ji. Exploiting kernel sparsity and entropy for interpretable cnn compression. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2800–2809, 2019.

[23] Shaohui Lin, Rongrong Ji, Yuchao Li, Cheng Deng, and Xuelong Li. Toward compact convnets via structure-sparsity regularized filter pruning. *IEEE transactions on neural networks and learning systems*, 2019.

[24] Shaohui Lin, Rongrong Ji, Yuchao Li, Yongjian Wu, Feiyue Huang, and Baochang Zhang. Accelerating convolutional networks via global & dynamic filter pruning. In *International Joint Conference on Artificial Intelligence*, pages 2425–2432, 2018.

[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 *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2790–2799, 2019.

[26] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European Conference on Computer Vision*, pages 740–755. Springer, 2014.

[27] Hanxiao Liu, Karen Simonyan, and Yiming Yang. Darts: Differentiable architecture search. *arXiv preprint arXiv:1806.09055*, 2018.

[28] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015.

[29] Jian-Hao Luo, Jianxin Wu, and Weiyao Lin. Thinet: A filter level pruning method for deep neural network compression. In *IEEE International Conference on Computer Vision*, pages 5058–5066, 2017.

[30] Hyeonwoo Noh, Seunghoon Hong, and Bohyung Han. Learning deconvolution network for semantic segmentation. In *Proceedings of the IEEE international conference on computer vision*, pages 1520–1528, 2015.

[31] Eilen Nordlie, Marc-Oliver Gewaltig, and Hans Ekkehard Plesser. Towards reproducible descriptions of neuronal network models. *PLoS computational biology*, 5(8):e1000456, 2009.

[32] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 652–660, 2017.

[33] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in neural information processing systems*, pages 5099–5108, 2017.

[34] Mohammad Rastegari, Vicente Ordonez, Joseph Redmon, and Ali Farhadi. Xnor-net: Imagenet classification using binary convolutional neural networks. In *European Conference on Computer Vision*, pages 525–542. Springer, 2016.

[35] Joseph Redmon and Ali Farhadi. Yolo9000: better, faster, stronger. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7263–7271, 2017.

[36] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

[37] Dennis W Ruck, Steven K Rogers, Matthew Kabrisky, Mark E Oxley, and Bruce W Suter. The multilayer perceptron as an approximation to a bayes optimal discriminant function. *IEEE Transactions on Neural Networks*, 1(4):296–298, 1990.

[38] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015.

[39] 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.

[40] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1):267–288, 1996.

[41] Diwen Wan, Fumin Shen, Li Liu, Fan Zhu, Jie Qin, Ling Shao, and Heng Tao Shen. Tbn: Convolutional neural network with ternary inputs and binary weights. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 315–332, 2018.

[42] Xiaodi Wang, Baochang Zhang, Ce Li, Rongrong Ji, Jungong Han, Xianbin Cao, and Jianzhuang Liu. Modulated convolutional networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 840–848, 2018.

[43] Bin Yang, Wenjie Luo, and Raquel Urtasun. Pixor: Real-time 3d object detection from point clouds. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 7652–7660, 2018.

[44] Ruichi Yu, Ang Li, Chun-Fu Chen, Jui-Hsin Lai, Vlad I Morariu, Xintong Han, Mingfei Gao, Ching-Yung Lin, and Larry S Davis. Nisp: Pruning networks using neuron importance score propagation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 9194–9203, 2018.

[45] Barret Zoph and Quoc V Le. Neural architecture search with reinforcement learning. *arXiv preprint arXiv:1611.01578*, 2016.