

FD-MOBILENET: IMPROVED MOBILENET WITH A FAST DOWNSAMPLING STRATEGY

Zheng Qin, Zhaoning Zhang, Xiaotao Chen, Yuxing Peng

Science and Technology on Parallel and Distributed Laboratory,
National University of Defense Technology, Changsha, China

ABSTRACT

We present *Fast-Downsampling MobileNet (FD-MobileNet)*, an efficient and accurate network for very limited computational budgets (e.g., 10-140 MFLOPs). Our key idea is applying a fast downsampling strategy to MobileNet framework. In FD-MobileNet, we perform $32\times$ downsampling within 12 layers, only half the layers in the original MobileNet. This design brings three advantages: (i) It remarkably reduces the computational cost. (ii) It increases the information capacity and achieves significant performance improvements. (iii) It is engineering-friendly and provides fast actual inference speed. Experiments on ILSVRC 2012 and PASCAL VOC 2007 datasets demonstrate that FD-MobileNet consistently outperforms MobileNet and achieves comparable results with ShuffleNet under different computational budgets, for instance, surpassing MobileNet by 5.5% on the ILSVRC 2012 top-1 accuracy and 3.6% on the VOC 2007 mAP under a complexity of 12 MFLOPs. On an ARM-based device, FD-MobileNet achieves $1.11\times$ inference speedup over MobileNet and $1.82\times$ over ShuffleNet under the same complexity.

Index Terms— Computer vision, convolutional neural network, deep learning

1. INTRODUCTION

Deep convolutional neural networks (CNNs) have become one of the most important methods in computer vision tasks such as image classification [1, 2, 3, 4], object detection [5, 6, 7, 8] and semantic segmentation [9, 10]. However, state-of-the-art CNNs require enormous computational resources and huge model sizes, which prevents them from being deployed on mobile or embedded devices.

For this reason, the inference-time compression and acceleration of deep neural networks has attracted the attention of the deep learning community in recent years. The related work is conventionally categorized into four classes. *Tensor decomposition* methods [11, 12] factorize a convolutional layer into several smaller convolutional layers, which reduces the overall complexity and the number of parameters. This class of methods conventionally involve a low-rank estimation process and a fine-tuning process, leading to a slow training procedure. *Parameter quantization* methods [13, 14] propose to utilize low-precision parameters in neural networks and provide significant theoretical speedup and enormous memory savings. However, current hardware is not well optimized for low-precision computation so specific hardware is required for quantization methods to achieve an ideal speedup. *Network pruning* methods [15, 16] attempt to discover and alleviate parameter and structure redundancy in deep neural networks. Early pruning approaches adopt an unstructured pruning scheme and induces random memory accesses, which is not

well supported by current hardware. Recent research on network pruning mainly focuses on structured pruning to leverage existing hardware. At last, *compact networks* [17, 18, 19] are specifically designed to employ both accurate and computationally economical networks on mobile or embedded devices.

Unlike the other methods which are mainly focused on compressing pre-trained models, compact networks can be trained from scratch. Additionally, compact networks are orthogonal to the other methods and can be further accelerated. In view of these advantages, various compact network architectures have been proposed. Among these networks, MobileNet [18] and ShuffleNet [19] achieve the state-of-the-art performance.

ShuffleNet is composed of a variant of the bottleneck unit [3] named the ShuffleNet unit. The ShuffleNet unit utilizes bypass connections for better representation capability. Beneficial from the powerful ShuffleNet unit, ShuffleNet achieves significant performance improvements over previous architectures [17, 18]. However, the bypass connection structure introduces multiple information paths in the computing graph, which induces frequent memory/cache switches in the engineering implementation on mobile or embedded devices. Consequently, the actual inference speed of ShuffleNet on physical devices is not ideal.

On the contrary, MobileNet exploits depthwise separable convolutions as its building blocks in a simple stacking architecture. This design allows a more efficient utilization of memory and cache, and MobileNet is significantly faster than ShuffleNet in actual inference speed under the same complexity. However, MobileNet adopts a slow downsampling strategy, which induces severe performance degradation when the computational budget is relatively small, for instance, 10-140 MFLOPs. In such a slow downsampling strategy, more layers have large feature maps, so the feature representation is more detailed. However, the number of channels in the network is restricted, thus the information capacity is relatively small. If the width of a network is further shrunk to fit an extremely limited complexity, the information capacity will become too small and the performance of the network will collapse.

In this paper, we present a highly efficient and accurate network named *Fast-Downsampling MobileNet (FD-MobileNet)* for extremely limited computational resources (e.g., 10 to 140 MFLOPs). Instead of merely shrinking the width of the network to fit small computational budgets, we compose FD-MobileNet by adopting a fast downsampling strategy into the MobileNet framework. In the proposed FD-MobileNet, we perform $32\times$ downsampling within the first 12 layers, which is only half of the number in the original MobileNet. After that, a sequence of depthwise separable convolutions are applied for better representation capability. Benefiting from the fast downsampling strategy, FD-MobileNet has the following three advantages: (i) The computational cost of FD-MobileNet is reduced as the spatial dimensions of the feature maps are smaller. (ii) FD-MobileNet allows more channels

Correspondence should be addressed to Zhaoning Zhang (Email: zznngxp@gmail.com).

than the MobileNet counterpart under the same complexity. This remarkably increases the information capacity of FD-MobileNet, which is critical to the performance of very small networks. (iii) FD-MobileNet inherits the simple architecture from MobileNet and provides a fast inference speed in engineering implementation.

We conduct extensive experiments to examine the effectiveness of the proposed FD-MobileNet. Firstly, we compare FD-MobileNet with other state-of-the-art compact networks on the ILSVRC 2012 dataset [20]. Then, we examine the generalization ability of FD-MobileNet on the PASCAL VOC 2007 dataset [21]. Experiments show that the proposed FD-MobileNet significantly outperforms MobileNet and achieves comparable performance with ShuffleNet under various computational budgets. For instance, FD-MobileNet achieves improvements of 5.5% on the ILSVRC 2012 top-1 accuracy and 3.6% on the VOC 2007 mAP over MobileNet under the computational budget of 12 MFLOPs. At last, we furthermore evaluate the actual inference speed of FD-MobileNet on an ARM-based device. Under a complexity of 12 MFLOPs, FD-MobileNet provides $1.11\times$ speedup over MobileNet and $1.82\times$ over ShuffleNet. Our code will be made publicly available later.

2. FAST-DOWNSAMPLING MOBILENET

In this section, we present the design of *Fast-Downsampling MobileNet (FD-MobileNet)*. FD-MobileNet is composed of the highly efficient depthwise separable convolutions and adopts a fast downsampling strategy. Benefiting from this design, FD-MobileNet achieves both high accuracy and high efficiency under very limited computational budgets.

Depthwise Separable Convolutions. Following MobileNet [18], FD-MobileNet exploits depthwise separable convolutions [22] as the building blocks. A $k \times k$ depthwise separable convolution factorizes a $k \times k$ standard convolution into a $k \times k$ depthwise convolution and a pointwise convolution with 8~9 times reduction in FLOPs. In practice, depthwise separable convolutions can achieve comparable performance with standard convolutions while provide great efficiency on computation-limited devices.

Fast Downsampling Strategy. Modern CNNs adopt a hierarchical architecture, where the spatial dimensions of the layers within the same stage is kept identical, and the spatial dimensions in the next stage is reduced by downsampling. In view of the restricted computational budgets, compact networks suffer from both the weak feature representation and the restricted information capacity. Different downsampling strategies provide a trade-off between detailed feature representation and large information capacity for compact networks. In a *slow downsampling strategy*, downsampling is performed in the later layers of the network, thus more layers have large spatial dimensions. On the contrary, downsampling is performed at the beginning of the network in a *fast downsampling strategy*, which significantly reduces the computational cost. Consequently, given a fixed computational budget, a slow downsampling strategy is inclined to generate more detailed features, whereas a fast downsampling strategy can increase the number of channels and allows more information to be encoded.

When the computational budget is extremely small, the information capacity plays a more important role in the performance of a network. Conventionally, the number of channels is reduced to adapt a compact network architecture to a certain complexity. In the case where a slow downsampling scheme is adopted, the network becomes too narrow to encode adequate information, which induces severe performance degradation. For instance, under a complexity of 12 MFLOPs, the original MobileNet architecture only has 128 chan-

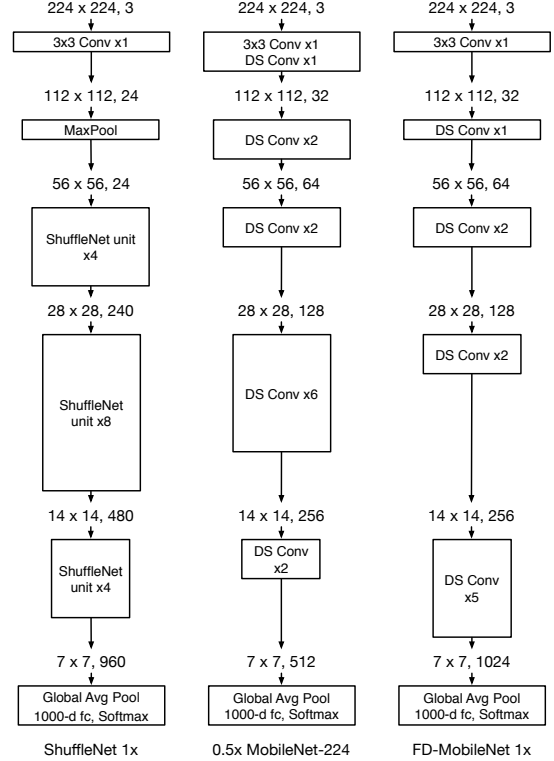


Fig. 1. Comparison of the downsampling strategies of FD-MobileNet, MobileNet and ShuffleNet under a complexity of 140 MFLOPs. The width of each block (except the ones at the bottom) represents the spatial dimensions while the height represents the number of building blocks. Compared with MobileNet and ShuffleNet, FD-MobileNet adopts a much faster downsampling strategy and leverages more channels, which enlarges the information capacity and gains performance improvements. **DS Conv**: depthwise separable convolution. Each depthwise separable convolution consists of two layers while each ShuffleNet unit has three layers.

nels in the last layer before the global pooling, thus the information capacity is very limited.

Based on this insight, we propose to adopt a fast downsampling strategy in the architecture of FD-MobileNet and postpone the feature extraction process to the smallest resolution. The faster downsampling is implemented by consecutively applying depthwise separable convolutions with large strides at the beginning of the network. Here we do not use max pooling because we find it does not gain performance improvements but introduces extra computation. The proposed FD-MobileNet accepts an image with a size of 224×224 pixels, and performs $4\times$ downsampling within the first 2 layers while performs $32\times$ downsampling within merely 12 layers, whereas the number of layers performing the same downsampling in the original MobileNet is 4 and 24, respectively. More specifically, the 12 layers are composed of 1 standard convolutional layer, 5 depthwise separable convolutions (each has a depthwise convolutional layer and a pointwise convolutional layer), and 1 depthwise convolutional layer. Fig. 1 illustrates the comparison of the downsampling strategies of FD-MobileNet, MobileNet and ShuffleNet under the computational budget of 140 MFLOPs. From the figure, it is observed that FD-MobileNet is significantly shallower than the

Table 1. Fast-Downsampling MobileNet architecture. “/2” indicates the stride of the layer is 2. **DWConv**: depthwise convolution.

Output Size	Layer	MFLOPs
224 × 224	Image	
112 × 112	3 × 3 Conv, 32, /2	10.8
56 × 56	3 × 3 DWConv, 32, /2	7.3
	1 × 1 Conv, 64	
28 × 28	3 × 3 DWConv, 64, /2	20.6
	1 × 1 Conv, 128	
	3 × 3 DWConv, 128	
	1 × 1 Conv, 128	
14 × 14	3 × 3 DWConv, 128, /2	19.9
	1 × 1 Conv, 256	
	3 × 3 DWConv, 256	
	1 × 1 Conv, 256	
7 × 7	3 × 3 DWConv, 256, /2	84.7
	1 × 1 Conv, 512	
	4 × 3 × 3 DWConv, 512	
	1 × 1 Conv, 512	
	3 × 3 DWConv, 512	
1 × 1	1 × 1 Conv, 1024	1.0
	Global Average Pooling	
	1000-d fc, Softmax	

other architecture before the feature maps are shrunk to 7×7 .

Remaining Layers. The utilization of the fast downsampling strategy significantly reduces the computation cost of the layers before the smallest spatial dimensions (7×7). Under the computational budget of 140 MFLOPs, MobileNet spends about 129 MFLOPs on the largest 4 resolutions, whereas FD-MobileNet only spends about 59 MFLOPs, as shown in Table 1. Consequently, more layers and more channels can be leveraged in the proposed architecture. Here we exploit 6 depthwise separable convolutions to improve the representation capability of generated features. The output channels of the first 5 depthwise separable convolutions are 512, while the last one is 1024, which is twice the number in the MobileNet counterpart ($0.5 \times$ MobileNet-224). The increase in the number of channels contributes to larger information capacity, which is critical to the performance of the networks under extremely limited computational resources.

Overall Architecture. The overall architecture of FD-MobileNet is demonstrated in Table 1. FD-MobileNet adopts a simple stacking architecture with 24 layers, including 1 standard convolutional layer, 11 depthwise separable convolutions, and 1 fully-connected layer. Following [18], a batch normalization [23] and a ReLU activation is applied after each convolutional layer. To conveniently adapt FD-MobileNet to different computational budgets, we introduce a hyper-parameter α termed *width multiplier* as in [18] to uniformly adjust the width of FD-MobileNet. We use a simple notation “FD-MobileNet $\alpha \times$ ” to represent a network with a width multiplier α , and the network in Table 1 is denoted as “FD-MobileNet $1 \times$ ”.

Inference Efficiency. Current deep learning frameworks accomplish the inference of a neural network by building an acyclic computing graph. For mobile or embedded devices, memory and cache resources are limited. As a result, complicated computing graphs can induce frequent memory/cache switches, which slows down the actual inference speed. FD-MobileNet inherits the simple architecture of the original MobileNet, and there is only one information path in the computing graph. This makes FD-MobileNet very friendly to engineering implementation and efficient on physical devices.

Table 2. Top-1 Accuracy (% , *larger is better*) on ILSVRC 2012 dataset. We re-implement MobileNet under a complexity of 12 MFLOPs as no results are reported in [18]

Models	MFLOPs	Top-1 Acc.
ShuffleNet $1 \times$ [19]	137	65.9
$0.5 \times$ MobileNet-224 [18]	149	63.7
FD-MobileNet $1 \times$ (<i>ours</i>)	144	65.3
ShuffleNet $0.5 \times$ [19]	38	57.3
$0.25 \times$ MobileNet-224 [18]	41	50.6
FD-MobileNet $0.5 \times$ (<i>ours</i>)	40	56.2
ShuffleNet $0.25 \times$ [19]	13	46.7
$0.125 \times$ MobileNet-224 [18]	12	39.6
FD-MobileNet $0.25 \times$ (<i>ours</i>)	12	45.1

3. EXPERIMENTS

3.1. Results on ILSVRC 2012 dataset

We first evaluate the effectiveness of FD-MobileNet on the ILSVRC 2012 dataset [20]. The ILSVRC 2012 dataset is composed of 1.2 million training images and 50,000 validation images. In the experiments, the networks are trained on the training set using PyTorch [24] with four GPUs for 90 epochs. Following [3], the batch size is set to 256 and a momentum of 0.9 is used. The learning rate starts from 0.1 and decays by an order of magnitude every 30 epochs. As the networks are relatively small, a weight decay of $4e-5$ is utilized as recommended in [19]. For data augmentation, we adopt a slightly less aggressive multi-scale augmentation scheme without using color jittering. On evaluation, the center-crop top-1 accuracy rates on the validation set are reported. Each validation image is first resized with its shorter edge to 256 pixels, and then evaluated using the center 224×224 pixels crop. Table 2 demonstrated the comparison of the top-1 accuracy of FD-MobileNet, MobileNet and ShuffleNet under three computational budgets.

From the table, FD-MobileNet achieves substantial improvements over MobileNet under different computational budgets. It is observed that FD-MobileNet surpasses MobileNet by a margin of 1.6% under a complexity of 140 MFLOPs, and performs 5.6% and 5.5% better when the computational budget is 40 and 12 MFLOPs, respectively. It is noteworthy that FD-MobileNet provides significantly improvements over MobileNet when the computational budget is very small (e.g., 40 and 12 MFLOPs). We attribute these improvements to the effectiveness of the fast downsampling strategy in FD-MobileNet. The original MobileNet adopts a slow downsampling strategy, thus more layers have relatively large feature maps and are more computationally intensive. Consequently, MobileNet is relatively narrow to maintain computational efficiency, which limits the information capacity. On the other side, FD-MobileNet exploits a much faster downsampling strategy, which allows more channels to be leveraged and alleviates the information capacity degradation. For instance, under 12 MFLOPs, the last layer of MobileNet outputs only 128 channels, whereas the number in FD-MobileNet is doubled. The increase in the information capacity significantly improves the performance of FD-MobileNet.

Compared with ShuffleNet, FD-MobileNet achieves comparable or slightly worse results. We conjecture that these differences are owed to the effectiveness of the bypass connection structure of the ShuffleNet unit. The bypass connection structure has proven powerful in various computer vision tasks [3, 5, 8]. However, on low-power mobile or embedded devices, the bypass connection

Table 3. mAP (% , larger is better) and AP (% , larger is better) on PASCAL VOC 2007 test set (600× resolution).

Backbone	mAP	areo	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv
0.5× MobileNet-224 [18]	53.8	59.0	66.8	52.3	33.5	29.8	56.9	71.4	61.5	29.8	59.2	51.3	59.3	69.9	64.6	63.5	29.5	48.9	51.8	65.0	52.1
FD-MobileNet 1× (ours)	55.4	58.1	67.1	49.4	32.7	28.8	62.2	71.1	67.2	32.6	59.4	58.0	63.0	72.3	65.7	65.8	26.9	53.5	51.9	65.0	56.7
0.25× MobileNet-224 [18]	42.3	47.5	53.8	35.0	24.0	18.5	43.9	60.2	51.3	17.5	47.6	47.5	47.4	60.0	58.7	55.5	19.2	38.3	36.3	48.9	34.2
FD-MobileNet 0.5× (ours)	45.1	46.4	53.2	38.2	29.3	16.8	47.1	63.0	56.2	22.3	48.8	49.4	47.3	66.9	60.6	56.8	20.0	44.5	40.9	57.0	38.0
0.125× MobileNet-224 [18]	29.1	33.4	38.6	20.7	16.0	2.9	31.4	48.5	42.7	13.2	26.8	28.2	34.5	46.9	45.4	42.3	13.4	29.3	21.7	29.6	16.1
FD-MobileNet 0.25× (ours)	32.7	40.6	43.6	21.4	16.2	8.2	33.7	50.7	41.2	15.6	37.2	33.4	36.2	54.7	50.4	41.4	8.1	29.6	25.3	47.7	18.2

structure induces frequent memory/cache switches and harms the actual inference speed. On the contrary, the simple architecture of FD-MobileNet contributes to an efficient utilization of memory and cache. Details are discussed in Section 3.3.

3.2. Results on PASCAL VOC 2007 dataset

We furthermore conduct extensive experiments on PASCAL VOC 2007 detection dataset [21] to examine the generalization ability of the proposed FD-MobileNet. PASCAL VOC 2007 dataset consists of about 10,000 images split into three (train/val/test) sets. In the experiments, the detectors are trained on VOC 2007 trainval set, and the single-model results on VOC 2007 test set are reported. We adopt the Faster R-CNN detection pipeline [5] and compare the performance of FD-MobileNet and MobileNet on 600× resolution under three computational budgets (140, 40 and 12 MFLOPs). The detectors are trained for 15 epochs with a batch size of 1. The learning rate starts from 1e-3, and is divided by 10 every 5 epochs. The weight decay is set to 4e-5. Other hyper-parameter settings follow the original Faster R-CNN in [5]. During testing, 300 proposals are sent to the R-CNN subnet to generate the final predictions.

The comparison of the results are demonstrated in Table 3. It is observed that FD-MobileNet achieves significant improvements over MobileNet under different computational budgets. Under the computational budget of 140 MFLOPs, the FD-MobileNet detector surpasses the MobileNet detector by a margin of 1.6% on mAP. The gap is enlarged when the complexity is lower. When the complexity is restricted to 40 and 12 MFLOPs, FD-MobileNet outperforms MobileNet by 2.8% and 3.6% on mAP, respectively. More specifically, on single class results, FD-MobileNet performs better than MobileNet on most classes. From Table 3, FD-MobileNet provides more significant improvements over MobileNet when the computational budget is smaller. For instance, when the computational budget is 12 MFLOPs, FD-MobileNet achieves consistent improvements on the classes which are hard for MobileNet, such as bottle (5.3%), chair (2.4%) and boat (0.2%). These improvements have proven that FD-MobileNet have strong generalization ability for transfer learning.

3.3. Actual Inference Time Evaluation

To investigate the performance on physical devices, we further compare the actual inference time of FD-MobileNet, MobileNet and ShuffleNet on an ARM-based platform. The experiments are conducted using an optimized NCNN framework [25] on an i.MX 6 series CPU (single-core, 800 MHz).

Table 4 shows the inference time of the three compact networks under computational budgets of 140, 40 and 12 MFLOPs, respectively. Compared with MobileNet, FD-MobileNet achieves about 1.1× speedup over MobileNet under the three computational budgets. These improvements are attributed to the effectiveness of the fast downsampling architecture of FD-MobileNet. Compared with ShuffleNet, FD-MobileNet provides significantly faster inference speed. When the computational budgets are 140 and 40 MFLOPs,

Table 4. Actual inference time (ms, smaller is better) on an ARM-based device with NCNN.

Models	MFLOPs	Time
ShuffleNet 1× [19]	137	522.27
0.5× MobileNet-224 [18]	149	431.73
FD-MobileNet 1× (ours)	144	391.66
ShuffleNet 0.5× [19]	38	204.97
0.25× MobileNet-224 [18]	41	155.84
FD-MobileNet 0.5× (ours)	40	139.47
ShuffleNet 0.25× [19]	13	103.79
0.125× MobileNet-224 [18]	12	63.73
FD-MobileNet 0.25× (ours)	12	57.17

FD-MobileNet gains 1.33× and 1.47× speedup over ShuffleNet, respectively. The speedup is elevated under a complexity of 12 MFLOPs: FD-MobileNet is 1.82× faster than ShuffleNet. It is noteworthy that under 140 and 40 MFLOPs, the ShuffleNet models have fewer FLOPs than the FD-MobileNet counterparts, but they are much slower. This slowdown is caused by the inefficiency of the bypass connection structure of the ShuffleNet unit. On low-power devices, the bypass connection structure leads to frequent memory and cache switch, which slows down the actual inference speed. On the contrary, the simple stacking architecture allows FD-MobileNet to leverage memory and cache more efficiently, which contributes to a faster actual inference speed. These results indicate that FD-MobileNet is effective in actual mobile or embedded applications.

4. CONCLUSION

In this work, we present Fast-Downsampling MobileNet (FD-MobileNet), a highly efficient and accurate network for very limited computational budgets. FD-MobileNet is built by adopting a fast downsampling strategy in the state-of-the-art MobileNet framework. Compare with the original MobileNet, the utilization of the fast downsampling scheme allows more channels, which increases the information capacity of the network and contributes to significant performance improvements. Experiments on the ILSVRC 2012 classification dataset and the PASCAL VOC 2007 detection dataset show that FD-MobileNet consistently outperforms MobileNet under different computational budgets. Evaluations of the actual inference time demonstrate that FD-MobileNet achieves significant speedup over ShuffleNet on an ARM-based device under the same complexity. For future work, we plan to adopt the fast downsampling strategy in other compact networks such as ShuffleNet for better performance.

5. ACKNOWLEDGMENT

This work is supported by the National Key Research and Development Program of China (2016YFB1000100).

6. REFERENCES

- [1] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton, “Imagenet classification with deep convolutional neural networks,” in *Advances in neural information processing systems*, 2012, pp. 1097–1105.
- [2] Karen Simonyan and Andrew Zisserman, “Very deep convolutional networks for large-scale image recognition,” *arXiv preprint arXiv:1409.1556*, 2014.
- [3] 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*, 2016, pp. 770–778.
- [4] Gao Huang, Zhuang Liu, Kilian Q Weinberger, and Laurens van der Maaten, “Densely connected convolutional networks,” *arXiv preprint arXiv:1608.06993*, 2016.
- [5] 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*, 2015, pp. 91–99.
- [6] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg, “Ssd: Single shot multibox detector,” in *European conference on computer vision*. Springer, 2016, pp. 21–37.
- [7] Joseph Redmon and Ali Farhadi, “Yolo9000: better, faster, stronger,” *arXiv preprint arXiv:1612.08242*, 2016.
- [8] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick, “Mask r-cnn,” *arXiv preprint arXiv:1703.06870*, 2017.
- [9] 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*, 2015, pp. 3431–3440.
- [10] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille, “Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs,” *arXiv preprint arXiv:1606.00915*, 2016.
- [11] Max Jaderberg, Andrea Vedaldi, and Andrew Zisserman, “Speeding up convolutional neural networks with low rank expansions,” *arXiv preprint arXiv:1405.3866*, 2014.
- [12] Xiangyu Zhang, Jianhua Zou, Kaiming He, and Jian Sun, “Accelerating very deep convolutional networks for classification and detection,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 38, no. 10, pp. 1943–1955, 2016.
- [13] Vincent Vanhoucke, Andrew Senior, and Mark Z Mao, “Improving the speed of neural networks on cpus,” in *Proc. Deep Learning and Unsupervised Feature Learning NIPS Workshop*, 2011, vol. 1, p. 4.
- [14] Matthieu Courbariaux, Itay Hubara, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio, “Binarized neural networks: Training deep neural networks with weights and activations constrained to +1 or -1,” *arXiv preprint arXiv:1602.02830*, 2016.
- [15] 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*, 2015, pp. 1135–1143.
- [16] Zhuang Liu, Jianguo Li, Zhiqiang Shen, Gao Huang, Shoumeng Yan, and Changshui Zhang, “Learning efficient convolutional networks through network slimming,” *arxiv preprint*, vol. 1708, 2017.
- [17] Forrest N Iandola, Song Han, Matthew W Moskewicz, Khalid Ashraf, William J Dally, and Kurt Keutzer, “Squeezenet: Alexnet-level accuracy with 50x fewer parameters and < 0.5 mb model size,” *arXiv preprint arXiv:1602.07360*, 2016.
- [18] 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.
- [19] Xiangyu Zhang, Xinyu Zhou, Mengxiao Lin, and Jian Sun, “Shufflenet: An extremely efficient convolutional neural network for mobile devices,” *arXiv preprint arXiv:1707.01083*, 2017.
- [20] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al., “Imagenet large scale visual recognition challenge,” *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.
- [21] 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*, vol. 88, no. 2, pp. 303–338, 2010.
- [22] François Chollet, “Xception: Deep learning with depthwise separable convolutions,” *arXiv preprint arXiv:1610.02357*, 2016.
- [23] Sergey Ioffe and Christian Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift,” in *International Conference on Machine Learning*, 2015, pp. 448–456.
- [24] Ronan Collobert, Koray Kavukcuoglu, and Clément Farabet, “Torch7: A matlab-like environment for machine learning,” in *BigLearn, NIPS Workshop*, 2011, number EPFL-CONF-192376.
- [25] Hui Ni, “Ncnn: A high-performance neural network inference framework optimized for the mobile platform,” <https://github.com/Tencent/ncnn>, 2017.