Training CNNs with Low-Rank Filters for Efficient Image Classification

Yani Ioannou[1] Duncan Robertson[2] Jamie Shotton[2] Roberto Cipolla[1] Antonio Criminisi[2]

[1]University of Cambridge, [2]Microsoft Research, Cambridge

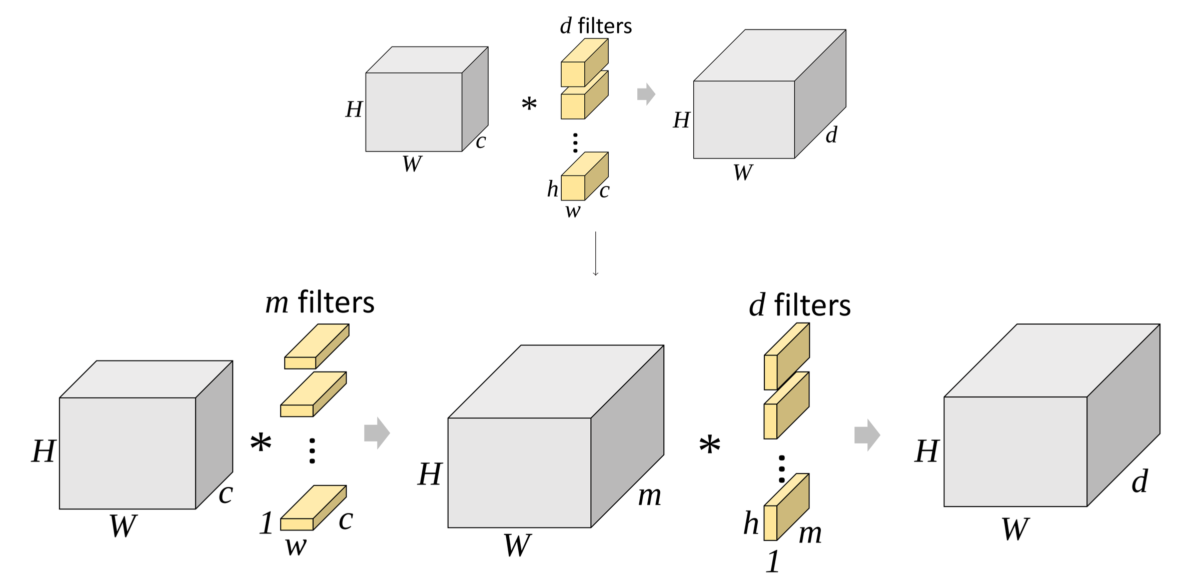
Microsoft

UNIVERSITY OF CAMBRIDGE

Summary

- We train CNNs with composite layers of oriented low-rank filters, of which the network learns the most effective linear combination
- In effect our networks learn a basis space for filters, based on simpler low-rank filters
- We propose an initialization for composite layers of heterogeneous filters, to train such networks from scratch
- Our models are faster and use less parameters
- With a small number of full filters, our models also generalize better

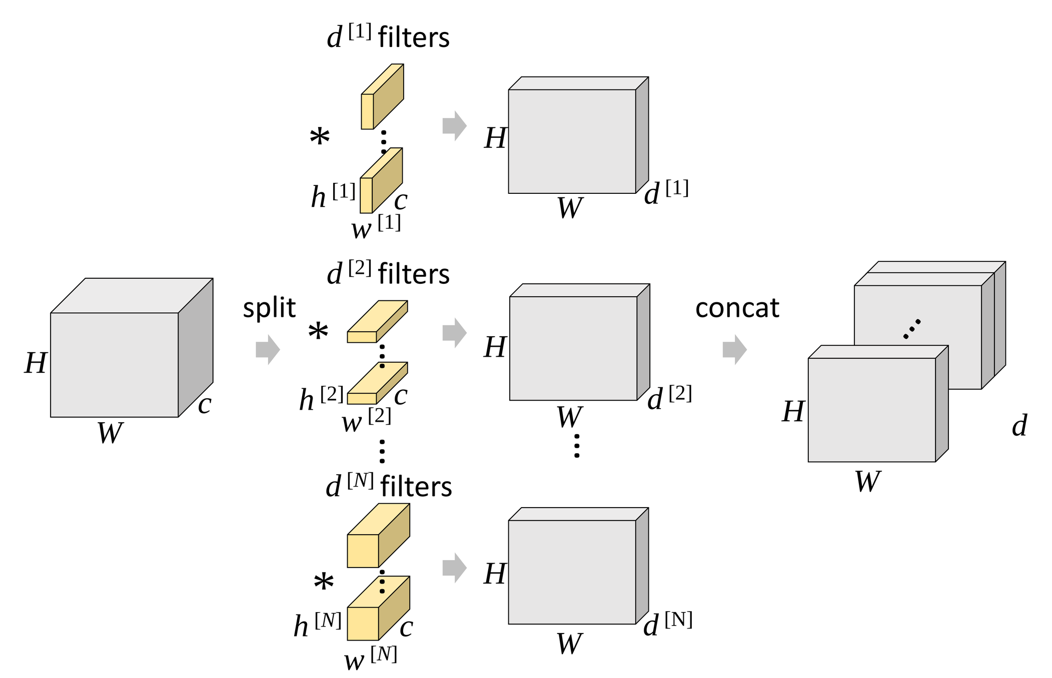
Previous Work: Separable (Factorized) Convolution



- Explicitly approximate low-rank factorization of trained CNN's full-rank filter
- Use sequential conv. layers with filters of differing orientation [3, 2].
- $\mathcal{O}(d\times[h\times w\times c]) \to \mathcal{O}(d\times[h\times \mathbf{m}]+\mathbf{m}[w\times c])$ (for each effective filter)
- However, in most CNNs, $d \geq m \gg c$, so this isn't much faster
- All previous methods approximated a pre-trained model!
- With our initialization, we can train these networks from scratch
- VGG-11 GMP Separable 88% top-5 accuracy on ILSVRC

Composite Layer - Initialization

- Incorrect initialization scales signal by β, for L layers, this becomes a scaling of β^L [1]
- If $\beta > 1$, $\beta^L \to \infty$, training diverges, if $\beta < 1$, $\beta^L \to 0$, training stalls

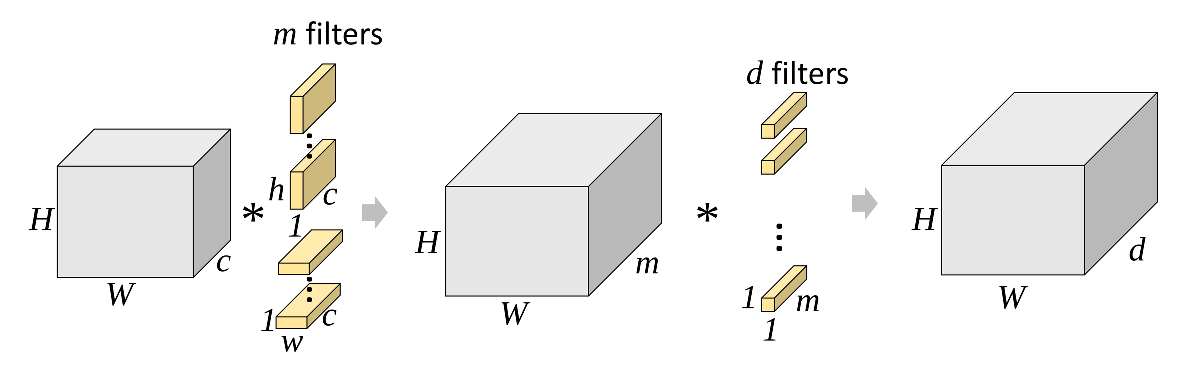


- When considering the initialization of composite layers (concatenated layers), must consider all layers for number of outgoing/incoming connections. For example, for a ReLU:

$$\sigma = \sqrt{\frac{2}{n_{\text{out}}}} = \sqrt{\frac{2}{\sum w^{[i]} h^{[i]} d^{[i]}}}.$$

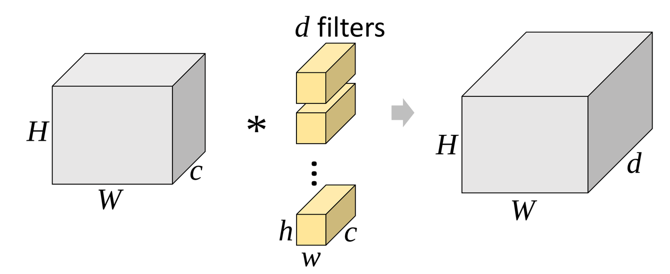
Proposed Method: Learning a Basis Space for Filters

- Intuition: Can a CNN learn to combine low-rank filters optimally during training?
- We structure our CNNs to learn a linear combination of low-rank filters, *i.e.* a basis space for filters:

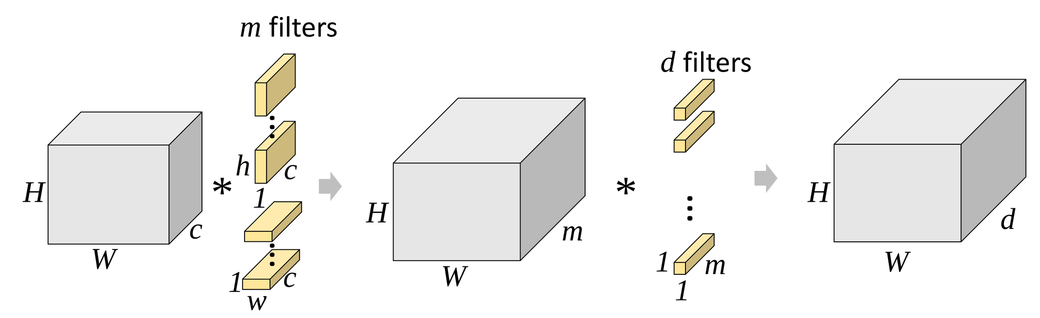


- A set of filters of different shape (similar to 'Inception', but low-rank and of different orientation [4])
- On the following layer, use $d \times [1\times 1\times m]$ filters to linearly combine
- No activation function between these two layers
- $\mathcal{O}(d\times[h\times w\times c]) \to \mathcal{O}(dm\times[h\times w\times c])$ (for each effective filter)

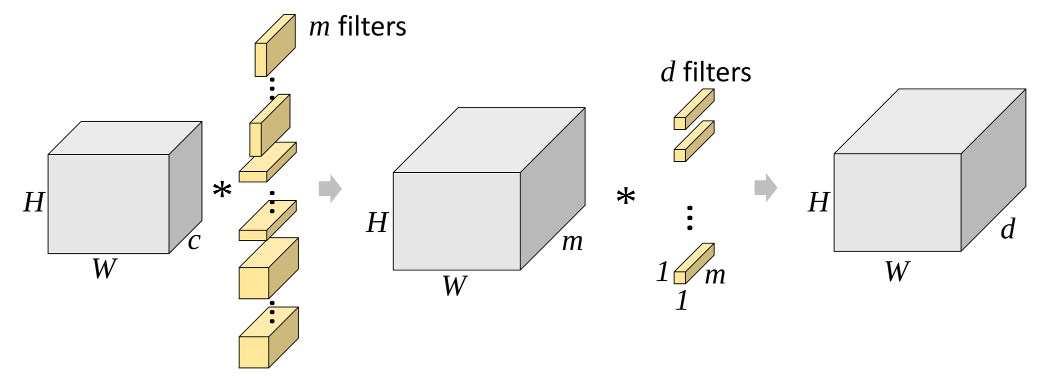
Training CNNs with Low-Rank Filters



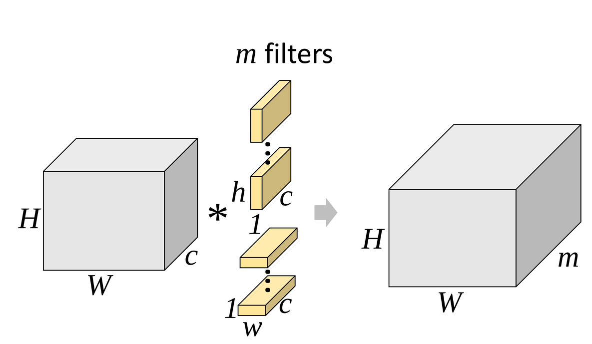
(a) A full rank convolutional layer.



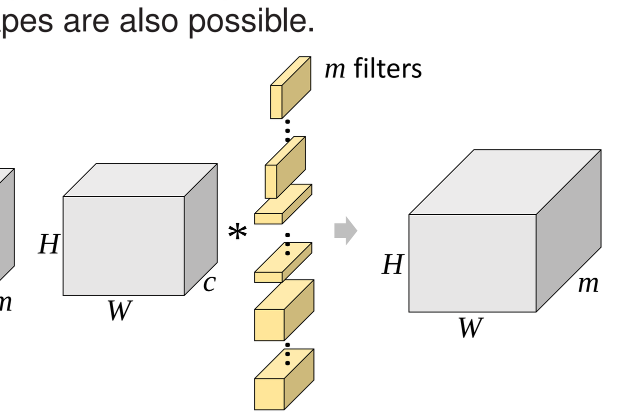
(b) Our method, a learned basis space of filters that are rectangular in the spatial domain and oriented horizontally and vertically.



(c) Our method, a learned basis space of rectangular filters and square filters. Filters of other shapes are also possible.



(d) Fig. (b), without explicit linear combination.



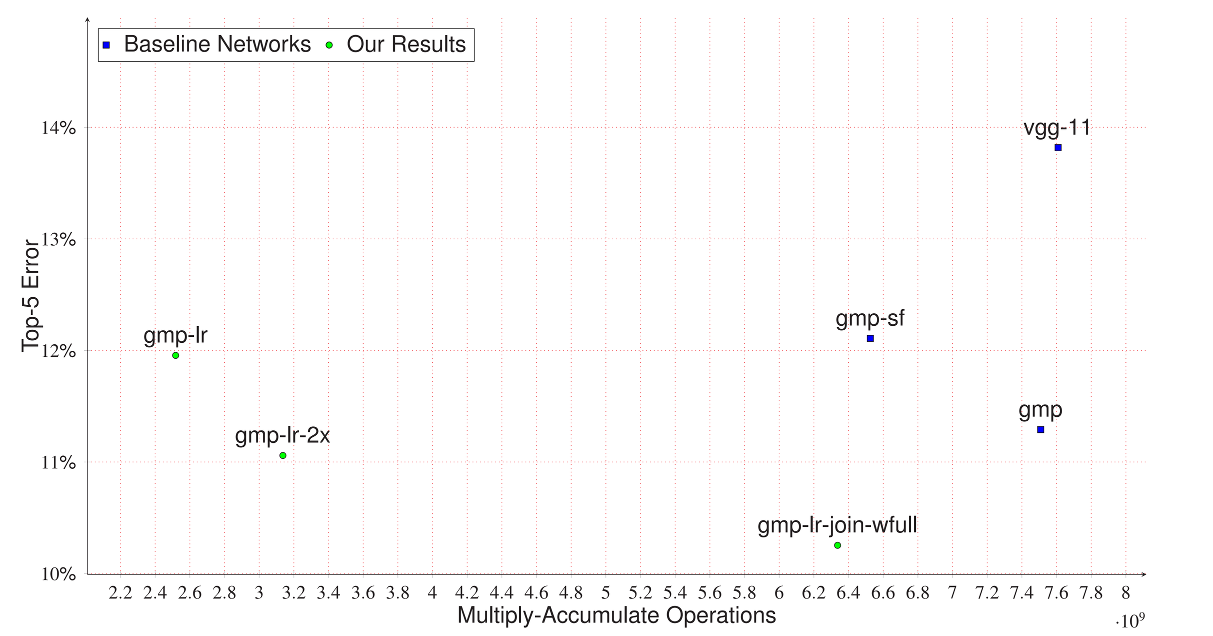
(e) Fig. (c), without explicit linear combination.

- Activation function is after the last layer in each configuration
- Shape of learned filters is a full $w \times h \times c$, but low-rank
- What can be effectively learned is limited by the number and complexity of the basis filters

VGG ILSVRC Results

Layer	VGG-11	GMP	GMP-SF	GMP-LR	GMP-LR-2X	GMP-LR-JOIN	GMP-LR-LDE	GMP-LR-JOIN-WFULL
conv1	3×3, 64		1×3, 64	3×1, 32 ‖ 1×3, 32	3×1, 64 ‖ 1×3, 64	3×1, 32 ‖ 1×3, 32		3×1, 24 ‖ 1×3, 24 ‖ 3×3, 16
			3×1, 64			1×1, 64	1×1, 32	1×1, 64
	ReLU							
	2×2 maxpool, /2							
conv2	3×3, 128		1×3, 128	3×1, 64 ‖ 1×3, 64	3×1, 128 ‖ 1×3, 128	3×1, 64 ‖ 1×3, 64		3×1, 48 ‖ 1×3, 48 ‖ 3×3, 32
			3×1, 128			1×1, 128	1×1, 64	1×1, 128
	ReLU							
	2×2 maxpool, /2							
conv3	3×3, 256		1×3, 256	3×1, 128 ‖ 1×3, 128	3×1, 256 ‖ 1×3, 256	3×1, 128 ‖ 1×3, 128		3×1, 96 ‖ 1×3, 96 ‖ 3×3, 64
			3×1, 256			1×1, 256	1×1, 128	1×1, 256
	ReLU							
	3×3, 256		1×3, 256	3×1, 128 ‖ 1×3, 128	3×1, 256 ‖ 1×3, 256	3×1, 128 ‖ 1×3, 128		3×1, 96 ‖ 1×3, 96 ‖ 3×3, 64
			3×1, 256			1×1, 256	1×1, 128	1×1, 256
	ReLU							
	2×2 maxpool, /2							
conv4	3×3, 512		1×3, 512	3×1, 256 ‖ 1×3, 256	3×1, 512 ‖ 1×3, 512	3×1, 256 ‖ 1×3, 256		3×1, 192 ‖ 1×3, 192 ‖ 3×3, 128
			3×1, 512			1×1, 512	1×1, 256	1×1, 512
	ReLU							
	3×3, 512		1×3, 512	3×1, 256 ‖ 1×3, 256	3×1, 512 ‖ 1×3, 512	3×1, 256 ‖ 1×3, 256		3×1, 192 ‖ 1×3, 192 ‖ 3×3, 128
			3×1, 512			1×1, 512	1×1, 256	1×1, 512
	ReLU							
	2×2 maxpool, /2							
conv5	3×3, 512		1×3, 512	3×1, 256 ‖ 1×3, 256	3×1, 512 ‖ 1×3, 512	3×1, 256 ‖ 1×3, 256		3×1, 192 ‖ 1×3, 192 ‖ 3×3, 128
			3×1, 512			1×1, 512	1×1, 256	1×1, 512
	ReLU							
	3×3, 512		1×3, 512	3×1, 256 ‖ 1×3, 256	3×1, 512 ‖ 1×3, 512	3×1, 256 ‖ 1×3, 256		3×1, 192 ‖ 1×3, 192 ‖ 3×3, 128
			3×1, 512			1×1, 512	1×1, 256	1×1, 512
	ReLU							
	2×2 maxpool, /2	global maxpool						
fc6	7^2 × 512 × 4096	512 × 4096						
	ReLU							
fc7	4096 × 4096							
	ReLU							
fc8	4096 × 1000							
	softmax							

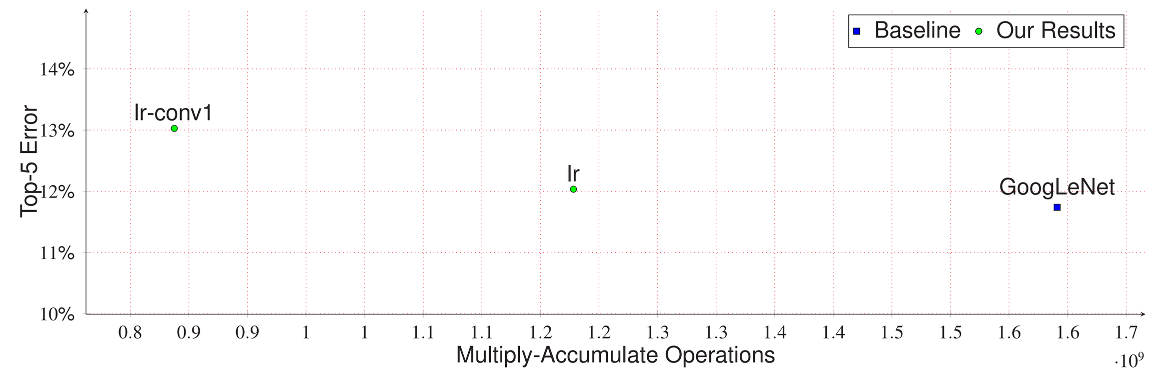
VGG Model Architectures. Here "3×3, 32" denotes 32 3×3 filters, "/2" denotes stride 2, fc denotes fully-connected, and ‖ denotes a concatenation within a composite layer.



Network	Stride	Multiply-Acc. $\times 10^9$	Param. $\times 10^7$	T1A	T5A
vgg-11	1	7.61	13.29	0.649	0.862
gmp	1	7.51	3.22	0.685	0.887
gmp-sf	1	6.53	2.97	0.673	0.879
gmp-lr-join-wfull	1	6.34	3.72	**0.704**	**0.897**
gmp-lr-join	1	3.85	2.73	0.675	0.880
gmp-lr-2x	1	3.14	3.13	0.693	0.889
gmp-lr	1	2.52	**2.61**	0.676	0.880
gmp-lr-lde	2	**1.02**	2.64	0.667	0.875

- Our models have significantly fewer FLOPS than the baseline network, in the case of 'gmp-lr-2x' by a factor of almost 60%, while slightly lowering error.
- The 'gmp-lr' and 'gmp-lr-join' networks have the same accuracy, showing that an explicit linear combination layer is unnecessary.
- Applying our method to an improved version of VGG-11 network using global max-pooling, we achieve comparable validation accuracy using 41% less compute and only 24% of the original VGG-11 model parameters.
- A mixture of full and low-rank filters gives a 1 percentage point *increase* in accuracy over our improved VGG-11 model, giving a top-5 *center-crop* validation accuracy of 89.7% while reducing computation by 16% relative to the original VGG-11 model.

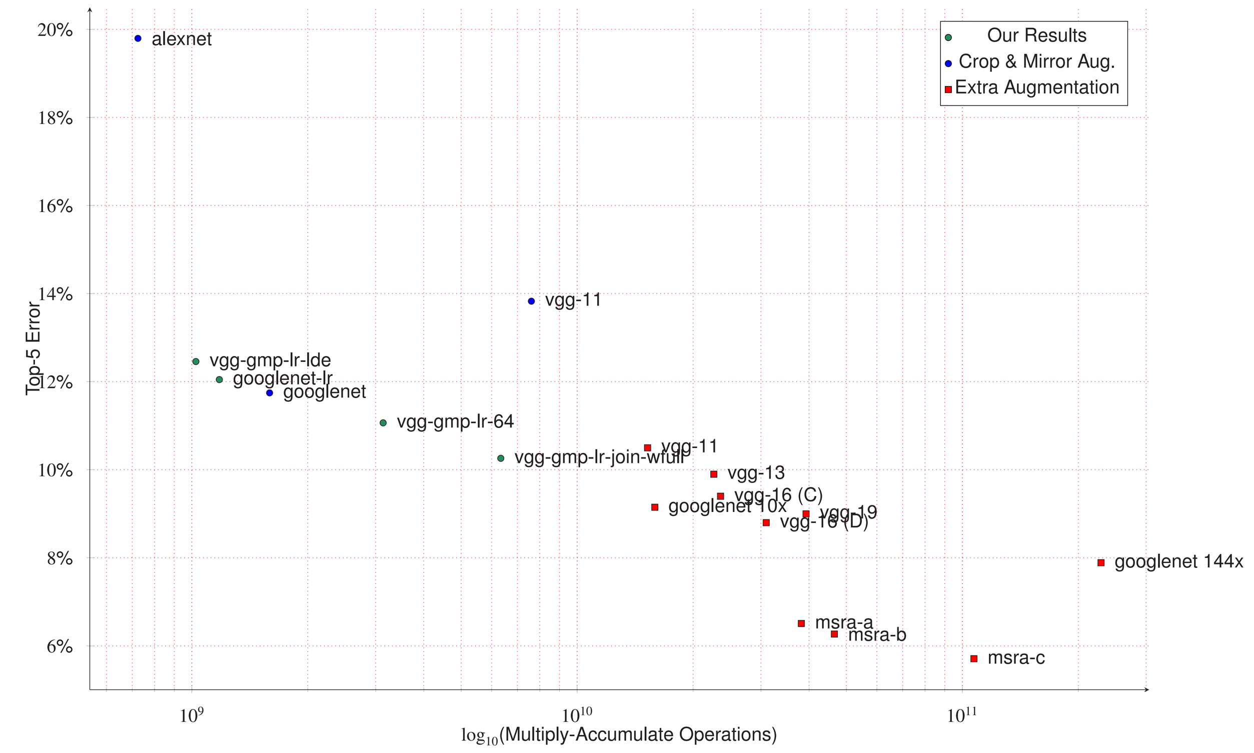
GoogLeNet ILSVRC Results



Network	Multiply-Acc. $\times 10^9$	Test Param. $\times 10^6$	T1A	T5A
GoogLeNet	1.59	5.97	**0.677**	**0.883**
lr	1.18	3.50	0.673	0.880
lr-conv1	**0.84**	**3.42**	0.659	0.870

- Applying our method to the optimized GoogLeNet architecture for ILSVRC, we achieved comparable accuracy with 26% less compute and 41% fewer model parameters.
- Google added similar low-rank filters with Inception v3 after our publication, showing an increase in accuracy with lower computation [5]

State-of-the-Art Models (at time of ICLR 2016 submission)



References

[1] Kaiming He et al. "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification". In: *Proceedings of the IEEE International Conference on Computer Vision*. 2015, pp. 1026–1034.

[2] Max Jaderberg, Andrea Vedaldi, and Andrew Zisserman. "Speeding up Convolutional Neural Networks with Low Rank Expansions." In: *British Machine Vision Conference*. 2014.

[3] Franck Mamalet and Christophe Garcia. "Simplifying convnets for fast learning". In: *Artificial Neural Networks and Machine Learning–ICANN 2012*. Springer, 2012, pp. 58–65.

[4] Christian Szegedy et al. "Going deeper with convolutions". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2015, pp. 1–9.

[5] Christian Szegedy et al. "Rethinking the Inception Architecture for Computer Vision". In: *arXiv preprint arXiv:1512.00567* (2015).

More Information