A. Appendices

A.1. Full Network-in-Network Results

Table 8: **Network-in-Network CIFAR10**

Model	FLOPS $\times 10^8$	Param. $\times 10^5$	Accuracy	CPU (ms)	GPU (ms)
Orig.	2.22	9.67	0.9211	39.0	0.623
root-2	1.64	7.37	0.9209	31.2	0.551
root-4	1.23	4.55	0.9202	27.6	0.480
root-8	1.03	3.15	**0.9215**	24.4	0.482
root-16	0.93	2.45	0.9167	23.0	0.475
tree-2	1.48	4.88	0.9185	31.4	0.541
tree-4	1.15	3.31	0.9147	29.1	0.535
tree-8	0.99	2.53	0.9171	25.7	0.500
tree-16	**0.91**	**2.14**	0.9168	20.6	0.512
col-2	1.53	5.71	0.9197	28.8	0.568
col-4	1.18	3.73	0.9200	26.1	0.536
col-8	1.01	2.73	0.9192	23.0	0.475
col-16	0.92	2.24	0.9120	22.8	0.494

Table 8 shows the full results for the Network-in-Network experiments on CIFAR10 on various hierarchical network topologies.

A.2. Inter-Layer Covariance

To show the relationships between filters between adjacent convolutional layers, as illustrated in Fig. 1 , we calculate the covariance of the responses from two adjacent featuremaps, the outputs of convolutional layers with c_1 and c_2 filters.

Let $X_i = [\mathbf{x}_{i,1}; \mathbf{x}_{i,2}; \ldots; \mathbf{x}_{i,N}]$ be the matrix of N samples $\mathbf{x}_{i,n}$ from the c_i dimensional featuremap for layer i. We consider each pixel across the two featuremaps to be a sample, and thus each vector $\mathbf{x}_{i,n}$ is a single pixel filter response of dimension c_i. If two featuremaps have different spatial dimensions, due to pooling, we up-sample the smaller featuremap (with nearest neighbor interpolation) such that there are the same number of pixels (and thus samples) in each featuremap.

Given two samples X_1, X_2 with zero mean (*i.e.* mean subtracted) for two adjacent featuremaps, we calculate the inter-layer covariance,

$$cov(X_1, X_2) = E\left[X_1 X_2^{\mathrm{T}}\right], \quad (1)$$

$$= \frac{1}{N-1} X_1 X_2^{\mathrm{T}}. \quad (2)$$

While this shows the covariance between layers, it is conflated with the inherent covariances within X_1 and X_2 from the data (as shown in Fig. 10a). We can more clearly show the covariance between layers by first whitening (using ZCA [22]) the samples in X_1 and X_2. For a covariance matrix,

$$cov(X, X) = \frac{1}{N-1} X X^{\mathrm{T}}, \quad (3)$$

The ZCA whitening transformation is given by,

$$W = \sqrt{N-1}\left(X X^{\mathrm{T}}\right)^{-\frac{1}{2}}. \quad (4)$$

Since the covariance matrix is symmetric, it is easily diagonalizable (*i.e.* PCA),

$$cov(X, X) = \frac{1}{N-1} X X^{\mathrm{T}}, \quad (5)$$

$$= \frac{1}{N-1} P D P^{\mathrm{T}}, \quad (6)$$

$$(7)$$

where P is a orthogonal matrix and D a diagonal matrix. This diagonalization allows a simplified calculation of the whitening transformation (see the derivation in Appendix A of [22]),

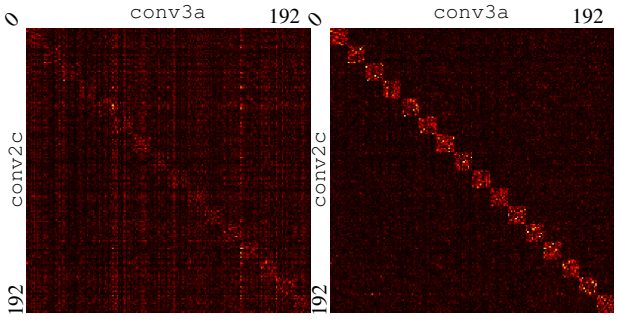
$$W = \sqrt{N-1} P D^{-\frac{1}{2}} P^{\mathrm{T}}, \quad (8)$$

where $D^{-\frac{1}{2}}$ is simply D with an element-wise power of $-\frac{1}{2}$.

The covariance between the whitened featuremap responses is then,

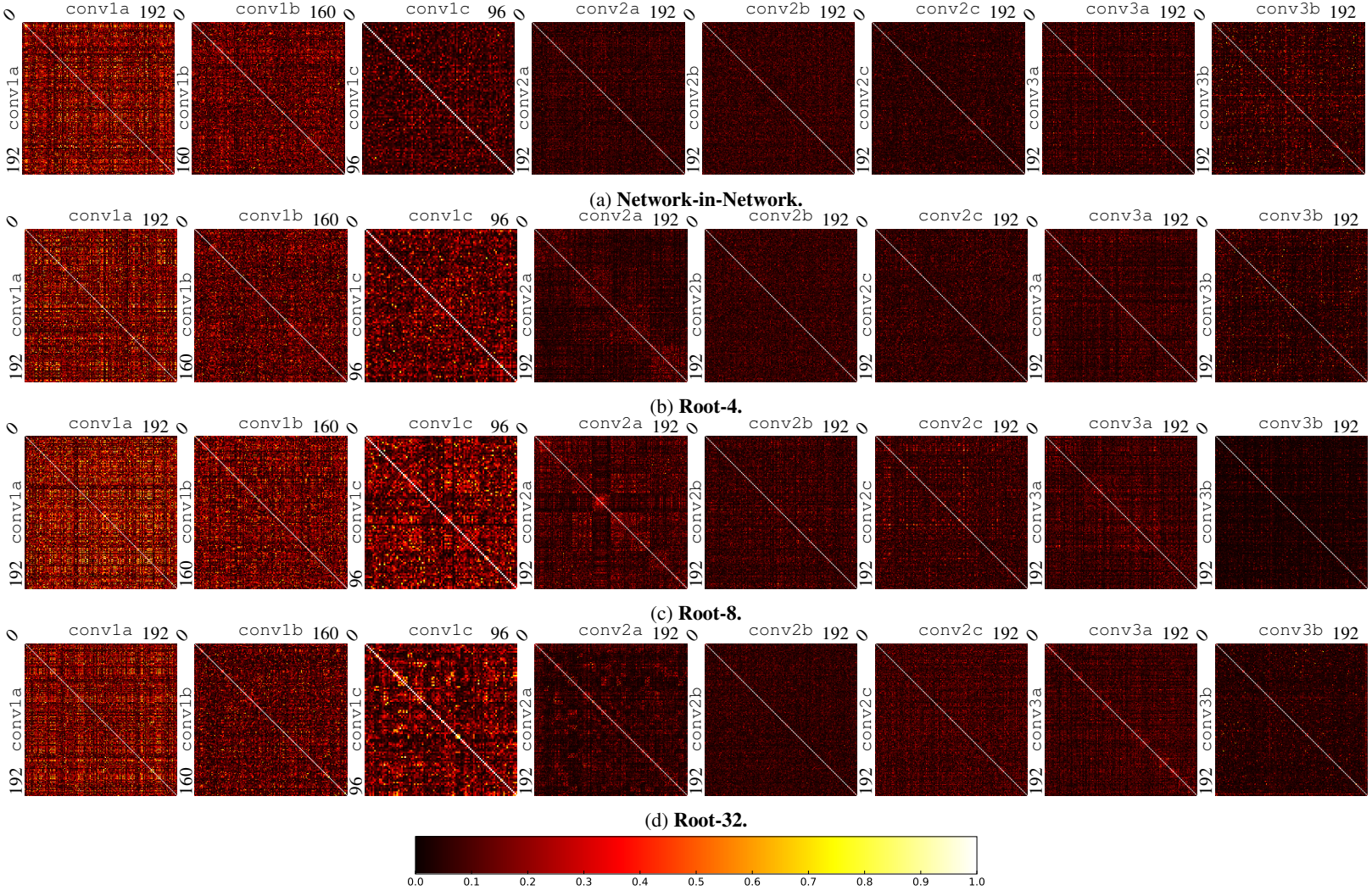
$$cov(W_1 X_1, W_2 X_2) = E\left[(W_1 X_1)(W_2 X_2)^{\mathrm{T}}\right]. \quad (9)$$

Figure 11 shows the per-layer (intra-layer) filter correlation. This shows the correlation of filters is more structured in root-networks, filters are learned to be linearly combined into useful filters by the root module, and thus filters are often grouped together with other filters they correlate strongly with.



(a) Non-whitened responses (b) Whitened responses

Figure 10: Covariance for between two layers in the root-32 Network-in-Network model with and without whitened responses



(a) **Network-in-Network.**

(b) **Root-4.**

(c) **Root-8.**

(d) **Root-32.**

Figure 11: **Network-in-Network Intra-Layer Correlation.** Absolute Correlation of filters within each layer of a NiN model variant.

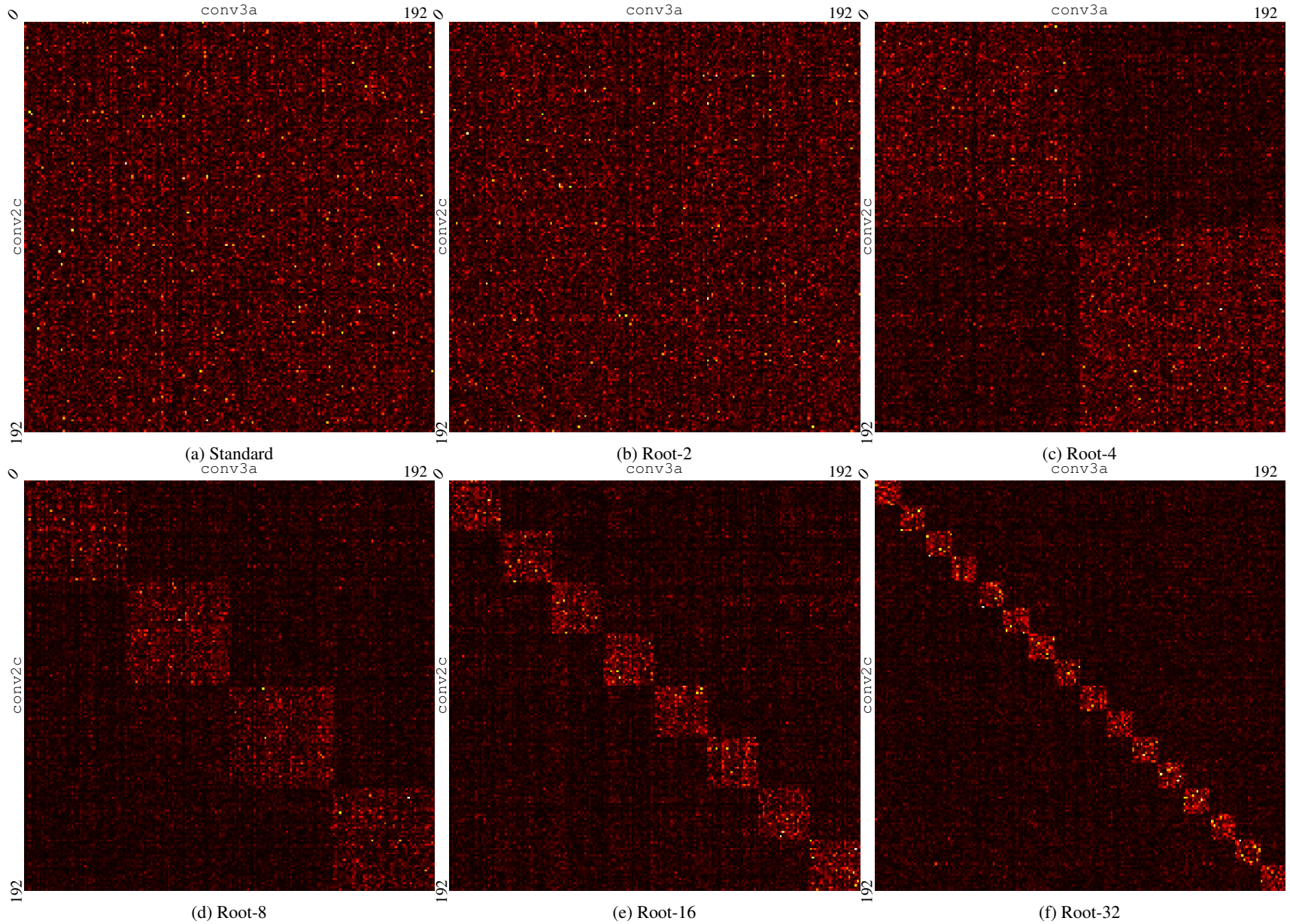
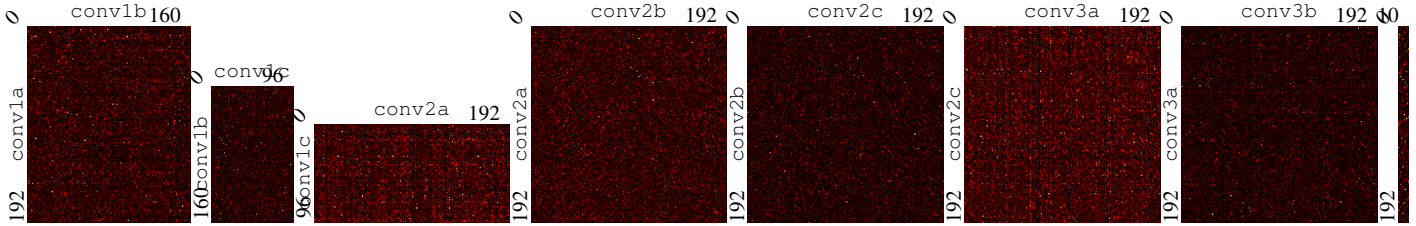
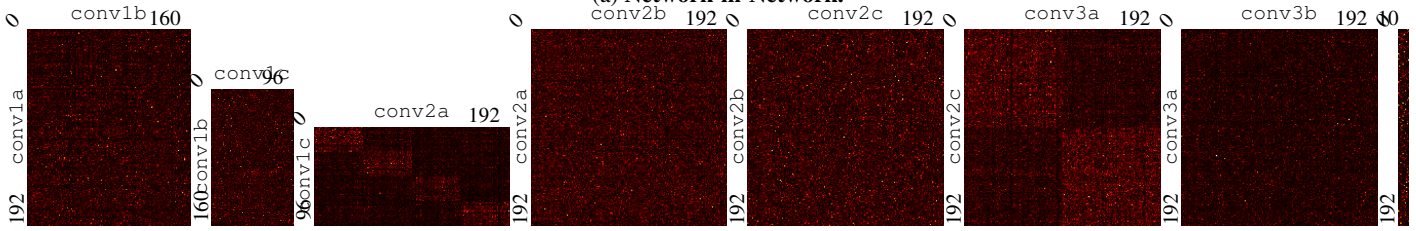


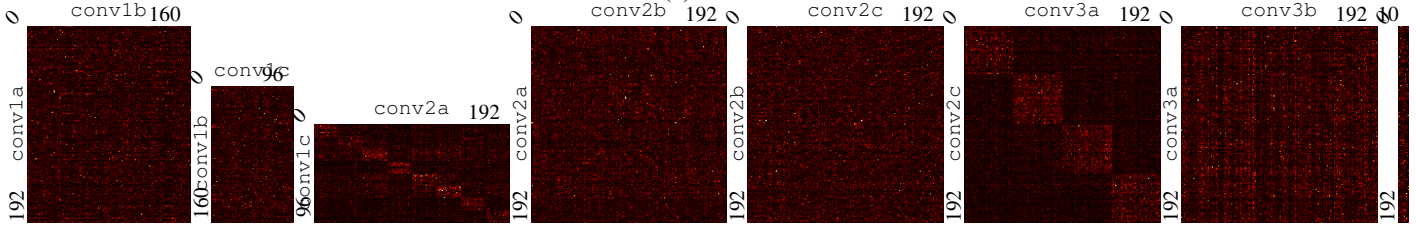
Figure 12: **Filter Inter-layer Covariance conv2c–conv3a.** The block-diagonal sparsity learned by a root-unit is visible in the correlation of filters on layers `conv3a` and `conv2c` in the NiN network.



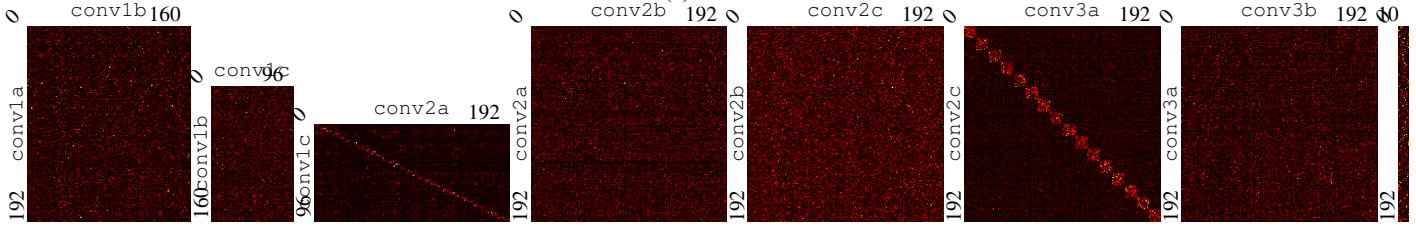
(a) **Network-in-Network.**



(b) **Root-4.**

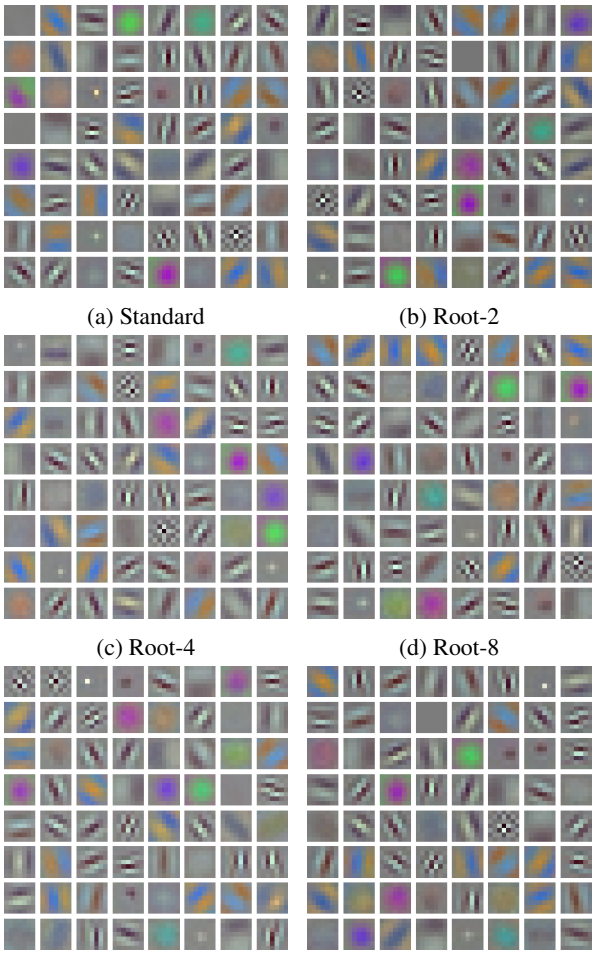


(c) **Root-8.**



(d) **Root-32.**

Figure 13: **Network-in-Network Inter-layer Absolute Covariance.** The inter-layer covariance for all layers in variants of the NiN network.



(a) Standard (b) Root-2

(c) Root-4 (d) Root-8

(e) Root-16 (f) Root-32

Figure 14: **ResNet 50 `conv1` filters.** With filter groups directly after `conv1`, in `conv2`, some of the organization of filters can be directly observed, and give us intuition as to what is happening in root networks.

Figure 12 shows the complete, enlarged version of Fig. 6, showing the inter-layer filter covariances between layers `conv3a` and `conv2c`. Figure 13 shows the full set of inter-layer covariances between all convolutional layers in the NiN models. Block-diagonal sparsity is visible on the layers with filter groups, `conv2a` and `conv3a`. This block-diagonal is shown for all variants in more detail in Fig. 13.

A.3. The Affect on Image-level Filters of Root Modules

In the ResNet root models, filter groups are used in `conv2`, directly after the image level filters of `conv1` some of the organization of filters can be directly observed, and give us intuition as to what is happening in root networks. Figure 14 shows the `conv0` filters learned for each of the ResNet 50 models. It is apparent that the filters learned in these networks are very similar to those learned in the original model, although sometimes inverted or with a different ordering. This ordering is somewhat consistent in models with filter groups however, even with different random initializations. This is because filter groups cause filters with strong mutual information to be grouped adjacent to each other.

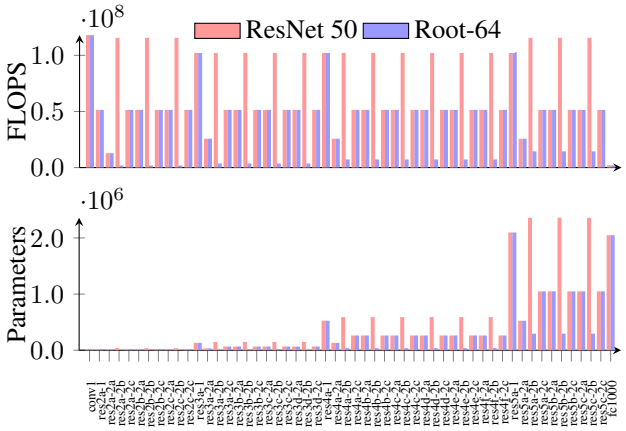


Figure 15: **ResNet 50 Layer-wise FLOPS/Parameters.**

For example, in the root-8 network (Fig. 14d), each row of filters corresponds to the input of an independent filter group in `conv2`. We can see that the first row primarily is composed of filters giving various directions of the same color gradient. These filters can be combined in the next layer to produce color edges easily. Due to the shortcut layer and the learned combinations of filters however, not all filter groupings are so obvious.

A.4. Layer-wise Compute/Parameter Savings

Figure 15 shows the difference in compute and parameters for each layer in a standard ResNet-50 model and a root-64 variant. The layers in the original networks with the highest computational complexity are clearly the spatial convolutional layers, *i.e.* layers with 3×3 spatial filters. When instead a root-module is used, the computational complexity of these layers is reduced dramatically. While the low dimensional embedding layers (1×1) are not changed, these have less than half the compute of the spatial convolution layers. The number of parameters in spatial convolution layers with large numbers of input channels, which increase towards the end of the network, are similarly reduced.