Spectral Leakage and Rethinking the Kernel Size in CNNs – Supplementary material –

Nergis Tomen, Jan C. van Gemert

A. Frequency resolution and scan of kernel sizes for the FFT regression task

As mentioned in Section 2.1, there are trade-offs between different window functions which should be taken into account in filter design. One of the main differences between using a conventional CNN kernel with the rectangular window, and using a standard tapering function is that a tapering function will effectively limit the size of the kernel in space domain. For example, once we taper a 7×7 kernel with the application of a standard window function, such as a Hamming window, the values of the weights close to the boundaries will be strongly attenuated, reducing the effective kernel size to a value more similar to 5×5 or 3×3 as shown in Fig. 6b. For a bandpass filter, the reduction of the window size in space domain leads to an increase in the width of the passband (or central lobe) in frequency domain via the uncertainty principle. This fundamentally decreases the frequency resolution attainable by the network for decreasing kernel size.

This effect is best demonstrated in the toy setup (Section 4.1) of the FFT regression experiment. In this setting, the minimum attainable regression loss (MSE loss) is limited by the maximum frequency resolution achievable by the network, which is determined by the kernel size. For the results presented in Section 4.1 (and shown in Fig. 4 and 5) we used a 7×7 kernel in the conventional convolutional layers and an 11×11 kernel in the layers using the Hamming window, in order to have a comparable frequency resolution between the two networks. However, it is not trivial to exactly match the frequency resolutions of such small, discrete kernels, and therefore, not easy to disentangle how much of the better performance in the Hamming model can be attributed to the frequency resolution and how much to artifact suppression. To survey the effect of the different kernel sizes, we perform a parameter scan of the kernel size in both networks. The resulting validation learning curves are shown in Fig. S.1.

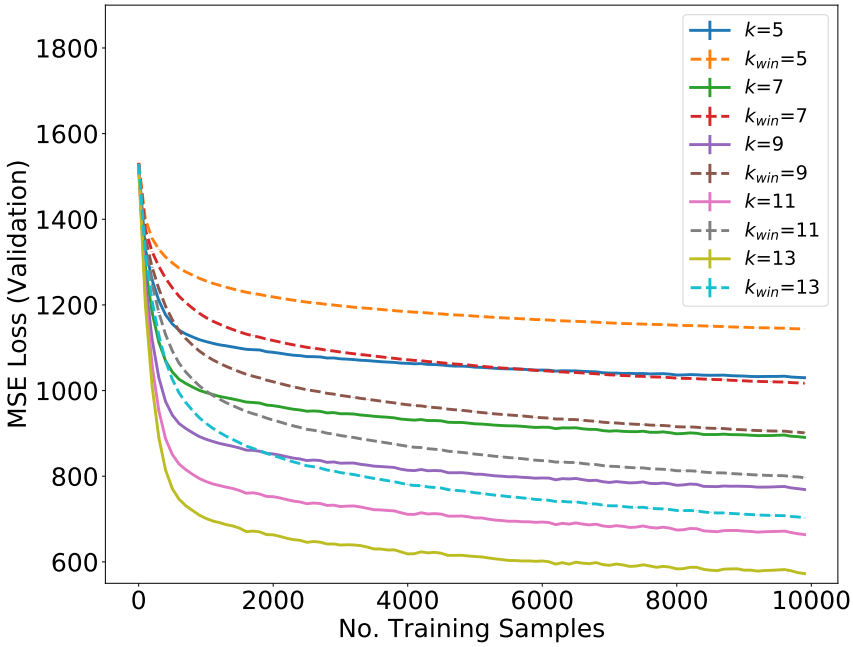

Figure S.1: Mean squared error loss in the baseline (solid lines) and Hamming (dashed lines) models with different kernel sizes on the FFT regression task. The loss is computed for an independent validation set of 1000 images. k and k_{win} refer to the kernel size in the baseline and Hamming models respectively. The setup of the experiments are identical to those described in Section 4.1 and to the learning curves shown in Fig. 5. The loss is averaged over 3 runs with random model initializations (standard deviation error bars are too small to see).

To give some insight into the effects the Hamming window has on the learned filters, we plot the kernels corresponding to different FFT frequency bins in both baseline and Hamming models after training (Fig. S.2). It is clear that very similar bandpass filters can be learned by both models, given large enough kernel sizes, however, the multiplication with a Hamming window in space domain has a tapering effect on the kernels in the Hamming models.

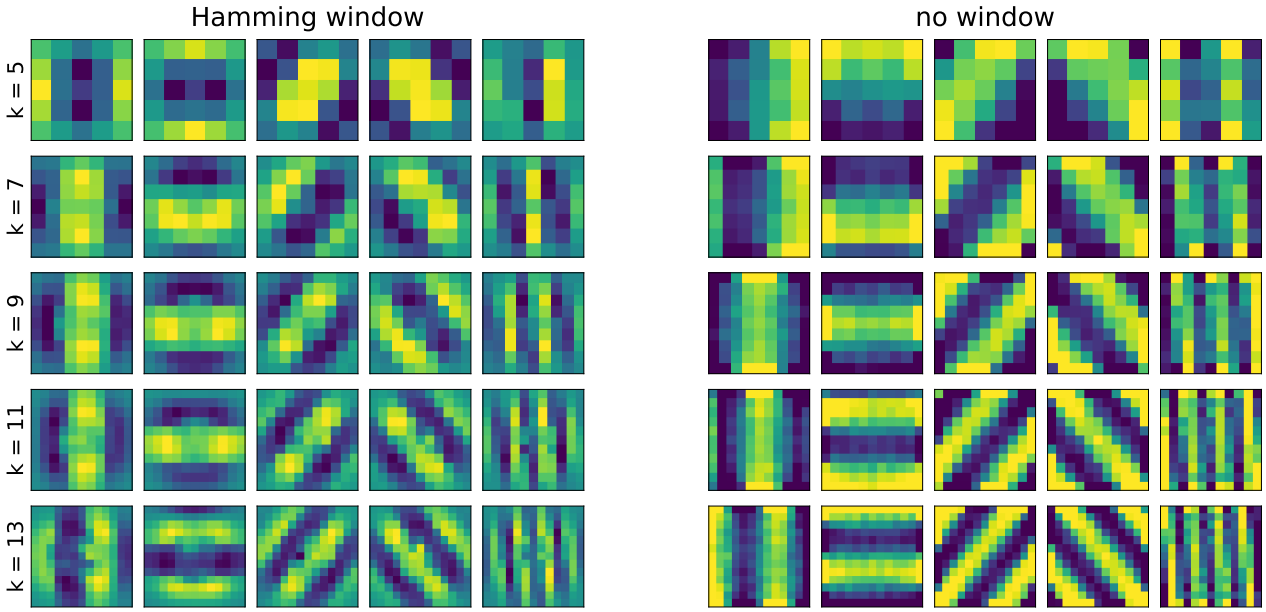

Figure S.2: Example filters learned in the FFT regression task by the Hamming models (left) and baseline models (right) with different kernel sizes k (rows). Each column corresponds to the kernel in one output channel of the convolutional layer, or equivalently, one frequency bin of the target FFT. We find that similar bandpass filters are learned in both models, however, the kernels are tapered of in space domain for the Hamming models.

The tapering performed by the Hamming window also works to attenuate spectral leakage in the frequency domain. This effect is demonstrated in the network predictions shown in Fig. S.3. We find that as the possible frequency resolution obtainable by the filters increases in both networks with increasing kernel size k, the networks are able to produce more precise predictions, with maximum responses located around the target frequencies. However, we also find that baseline networks, while optimizing their kernels to function as precise bandpass filters, also learn filters which are prone to considerable spectral leakage. This observation adds support to our original hypothesis that conventional CNN filters are susceptible to leakage artifacts, as the baseline networks in the FFT regression task still suffer from them, even when they directly contribute to the loss function. In contrast, the Hamming models are more successful in suppressing leakage artifacts, which indicates that explicit regularization by a window function may be useful in convolutional architectures.

B. Orthogonality in windowed convolutions

Orthogonal convolutional neural networks [3] have recently been proposed as an approach which can alleviate, to some degree, problems in CNNs related to overparameterization [2] or under utilization of model capacity [1]. Orthogonal representations may reduce redundancies in the learned weights and lead to better performance through optimal capacity usage. In [3], Wang et al. suggest that the correct, and effective, way to enforce orthogonality is not to impose orthogonality between individual kernels, but between the rows or columns of the doubly block-Toeplitz (DBT) matrix which defines the linear operation in a convolutional layer.

As explained in Section 4.4, spectral leakage may reduce the frequency-selectivity of the filters in a convolutional layer. Thus the network model may be more prone to learning redundant representations. We suggest that, by removing truncation artifacts, the Hamming window may enable the learning of more orthogonal filters, with less overlap in their frequency responses. Hence, CNNs with larger kernels with the Hamming window, may lead to both redundancy reduction, and performance increase, as well as to convolutional layers more similar to an orthogonal convolution.

In order to test this hypothesis, we construct the DBT matrix $\mathcal{K}$ using the weight tensor K in each layer. Specifically, for a convolutional layer with C input channels, M output channels and kernel size k, the standard weight tensor $K \in \mathbb{R}^{M\times C\times k\times k}$ can be rearranged, such that the linear operator of the convolutional layer can be written as a matrix multiplication between a flattened input feature map $\mathbf{x} \in \mathbb{R}^{CHW}$ and the DBT matrix $\mathcal{K} \in \mathbb{R}^{(MH'W')\times(CHW)}$

$$\mathbf{y} = \mathcal{K}\mathbf{x} \tag{S.1}$$

where $\mathbf{y} \in \mathbb{R}^{(MH'W')}$ is the output feature map before nonlinearities or normalization. H and W, and H' and W' denote the spatial dimensions of the input $\mathbf{x}$ and output $\mathbf{y}$, respectively.

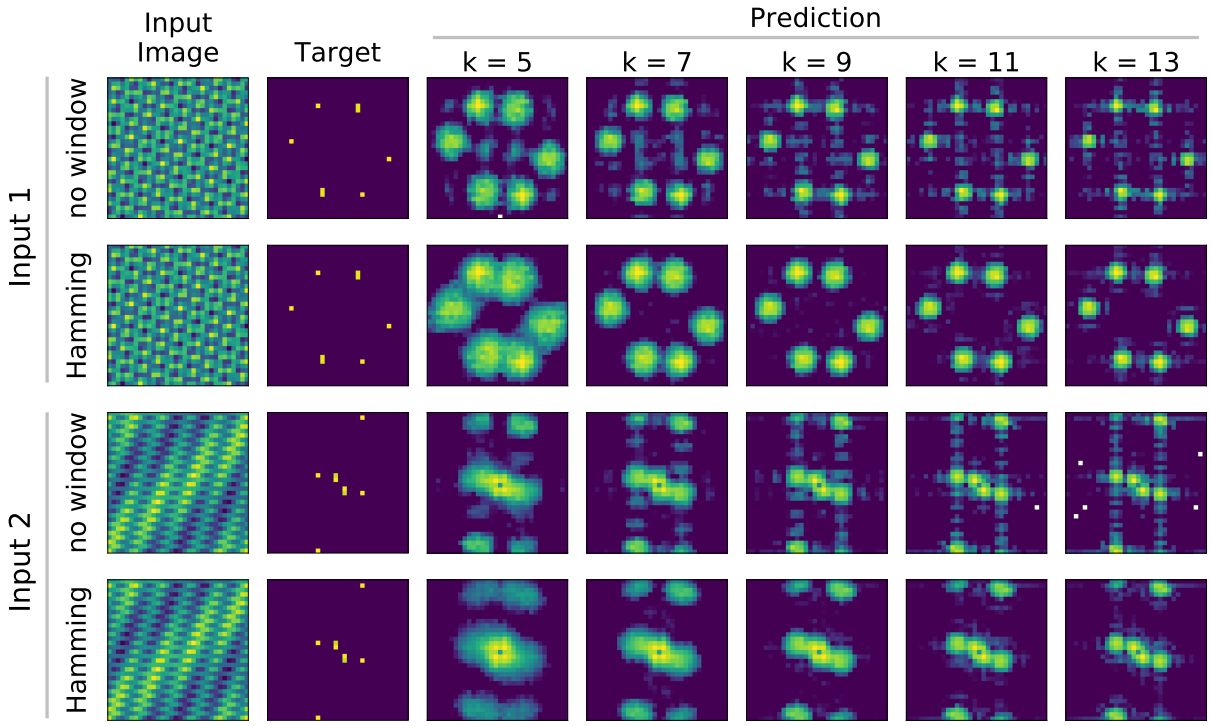

Figure S.3: Predictions of the baseline (no window) and Hamming networks with different kernel sizes k for two example input images. The input images and the corresponding target FFTs are the same as those shown in Fig. 4. We find that the responses of different output channels, and hence the predictions of the models, become more localized with increasing k, as the central lobe of the filter responses become narrower. However, we find that baseline networks do not readily suppress leakage artifacts, which can appear far from target frequency bins, even when they corrupt the predictions. Hamming models, in contrast, are more successful in keeping the responses constrained around target frequencies.

For our models, we consider the row orthogonality condition for the DBT matrix

$$\delta_{ij} = \langle \mathcal{K}_{i,.}, \mathcal{K}_{j,.} \rangle = \begin{cases} 1, & \text{if } i = j \\ 0, & \text{else} \end{cases} \tag{S.2}$$

which shows that, for an orthogonal convolution, each row vector $\mathcal{K}_{i,.}$, corresponding to the i-th row of the DBT matrix, must be orthogonal to every other row vector of the DBT matrix, so that every pairwise dot product δ_{ij} between the row vectors $\mathcal{K}_{i,.}$ and $\mathcal{K}_{j,.}$ is 0 except when $i = j$.

Based on this, we would like to find how much the ResNet and VGG models trained on ImageNet deviate from orthogonal convolutions in each layer. Therefore, we first normalize each row of the DBT matrix $\mathcal{K}_{i,.}$ to a unit vector, so that $\langle \mathcal{K}_{i,.}, \mathcal{K}_{i,.} \rangle = 1$, then compute the dot product $\hat{\delta}_{ij}$ between every pair of rows (i, j). We report the average deviation D of the dot product from the orthogonality condition

$$D = \frac{1}{N(N-1)} \sum_{i=1}^{N} \sum_{j \neq i} |\hat{\delta}_{ij}| \tag{S.3}$$

for every convolutional layer, where $N = MH'W'$ is the number of rows of the DBT matrix. In order to illustrate the effects of training, we also compute D for a randomly initialized version of each model, which we call the 'chance' level.

We find that the convolution operators deviate from orthogonality the least when using the Hamming window for all ResNet and VGG models S.4. For all baseline models, we find that training increases the deviation from orthogonality, and D is on average higher than its corresponding chance level. However, we also find that this difference between the deviation D of the trained model and its chance level is minimized for the Hamming models.

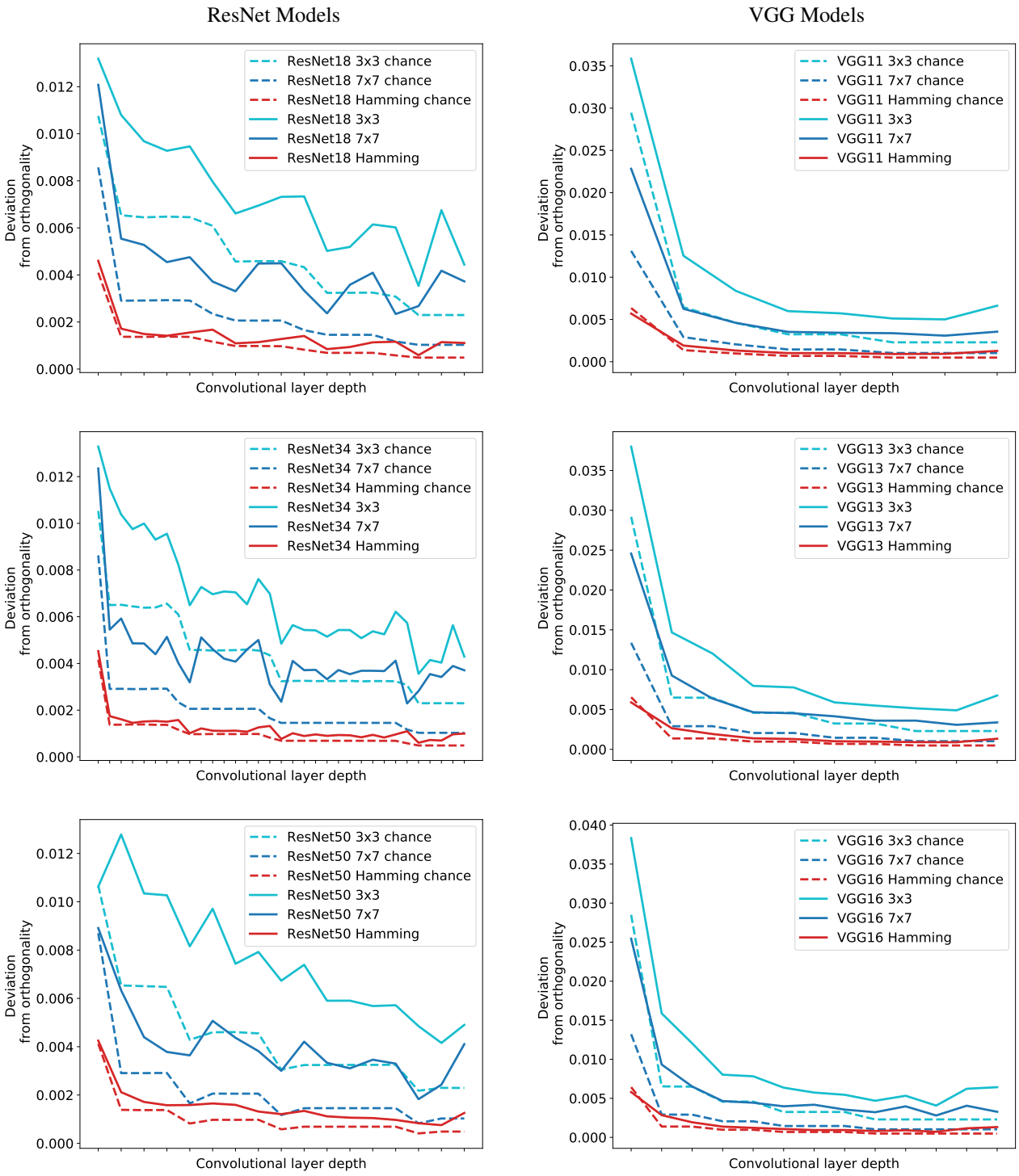

Figure S.4: Deviation of the convolutional layers in different ResNet and VGG models from orthogonal convolutions [3]. The deviation D is quantified as the average dot product between the row vectors of the doubly block-Toeplitz (DBT) matrix, as given in Eq. S.2-S.3. A 'chance' level is computed using a randomly initialized version of every model (dashed lines). We find that Hamming-windowed convolutions (red) deviate minimally from orthogonal convolutions as compared to baselines (cyan and blue). In addition, the increase in D caused by training is the smallest for Hamming models.

C. Training Loss

As discussed in Sections 2.3 and 3, we believe the Hamming window acts as a regularizer, preventing the network from overfitting to leakage artifacts. The regularization effect of the Hamming window seems to provide extra benefits to standard regularization methods, such as weight decay and data augmentation (Fig. 6a bottom right and Fig. 7a right).

However, to further confirm that the effect of the Hamming window is not simply avoiding overfitting to training data, here we present the training curves for the ImageNet and CIFAR-100 experiments.

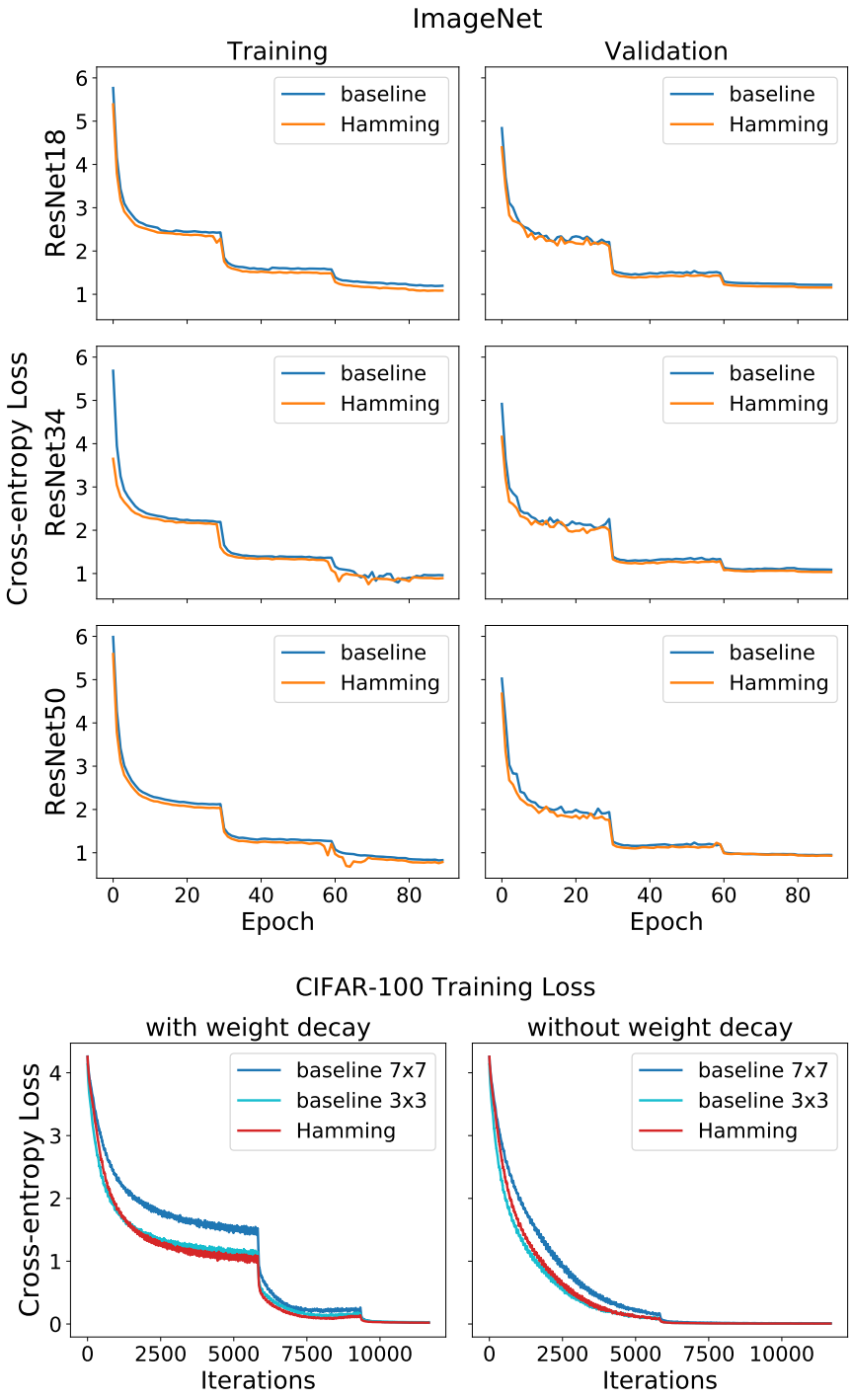

Figure S.5: Training and validation loss for the ImageNet experiments (top) and training loss in the CIFAR-100 experiments (bottom). ImageNet models correspond to the same models shown in Fig. 7c. CIFAR-10 models are $M = 12$ layer models trained for 150 epochs.

D. Weight decay

To reaffirm our observations that the effect of the windowed convolutions aren't cancelled out by weight decay regularization, we performed a weight decay sweep on the CIFAR-10 and CIFAR-100 models.

We use the $M = 6$ layer models for the CIFAR-10 experiments (the setup is identical to the setup of Fig. 6a, bottom right; i.e. all layers are windowed and we use the 'non-wide' channel setting), and we use the $M = 8$ layer models for the CIFAR-100 experiments (the setup is identical to that of Fig. 7a, right). All results are averaged over 5 runs.

We observe that the networks using the Hamming window outperform both baselines, with kernel sizes $k = 3$ and $k = 7$ respectively. We find that the validation accuracy is maximized at a weight decay rate of 0.0025 for all models. This shows that using the Hamming window provides a consistent improvement, and its effect cannot be compensated for by weight decay.

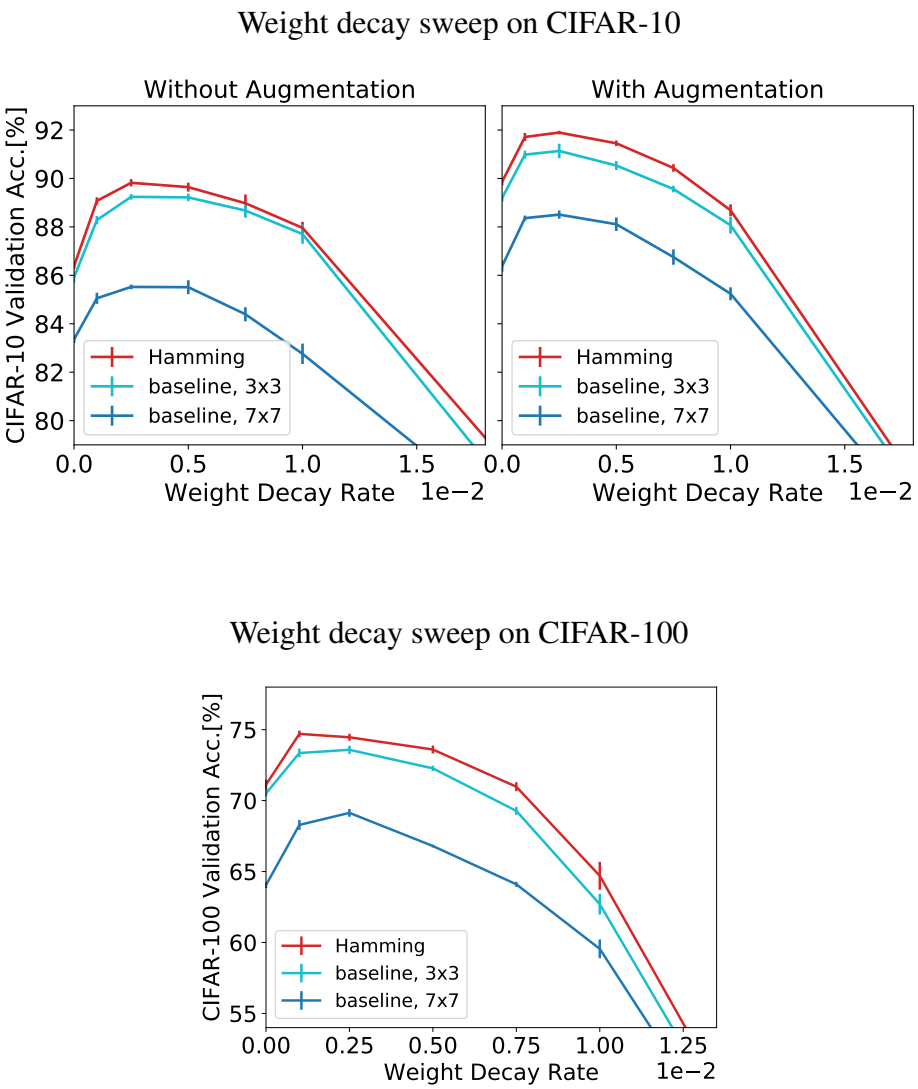

Figure S.6: CIFAR-10 (top) and CIFAR-100 (bottom) validation accuracy of models using the Hamming window with $k = 7$ (red) compared to baseline models without windowing with $k = 7$ (blue) and $k = 3$ (cyan). Error bars show the standard deviation over 5 runs.

References

[1] Brian Cheung, Alexander Terekhov, Yubei Chen, Pulkit Agrawal, and Bruno A. Olshausen. Superposition of many models into one. In *Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019*, pages 10867–10876, 2019. 2

[2] Song Han, Huizi Mao, and William J. Dally. Deep compression: Compressing deep neural network with pruning, trained quantization and huffman coding. In *4th International Conference on Learning Representations, ICLR 2016*, 2016. 2

[3] Jiayun Wang, Yubei Chen, Rudrasis Chakraborty, and Stella X. Yu. Orthogonal convolutional neural networks. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020*, pages 11502–11512. IEEE, 2020. 2, 4