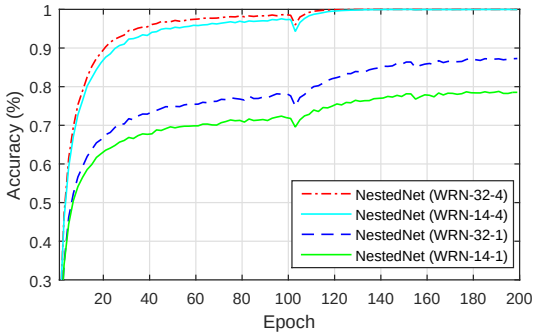
NestedNet: Learning Nested Sparse Structures in Deep Neural Networks [Supplementary Material]

Eunwoo Kim Chanho Ahn Songhwai Oh
Department of ECE and ASRI, Seoul National University, South Korea
{kewoo15, mychahn, songhwai}@snu.ac.kr

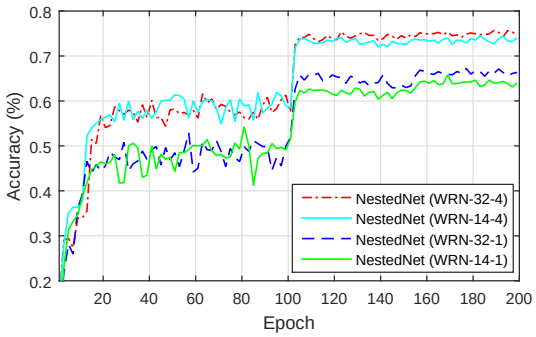
Table 1 describes the taxonomy of the Imagenet subset, named ImageNet-Subtree, which is performed for hierarchical classification in the main paper. We also provide performance curves and implementation details as well as activation maps for NestedNet in the following sections.

1. Performance Curves

We provide performance curves of NestedNet for train and test sets in CIFAR-100 while training on the knowledge distillation problem, where we use the same architecture to those used in the main paper. Train and test accuracies of each internal network while learning the nested network, which are computed in every epoch by averaging for all batch sets in train and test images, respectively, are shown in Figure 1. For the experiment, we have empirically found that the curves obtained from the n-in-1 nested sparse network, whose internal networks are learned simultaneously, give similar trend to those obtained from the independently learned baseline networks. Further details and results of the nested network are described in the main paper.



(a) Train



(b) Test

Figure 1. Performance of NestedNet while training on the CIFAR-100 dataset [4]. $(\cdot)$ denotes the applied architecture.

2. Implementation Details

2.1. CIFAR datasets

We implement NestedNets based on state-of-the-art networks such as residual networks (ResNet) [2] and wide residual networks (WRN) [6]. We follow the practice in [2] to construct those networks whose number of layers is $6n_b$+2, where n_b is the number of residual blocks. We initialize weights in all compared architectures using the Xavier initialization [1] and train them from scratch. For NestedNet, we use the SGD optimizer with momentum of 0.9 and the Nesterov acceleration method where the size of a mini-batch is 128. Batch normalization [3] is adopted after each convolutional operation and dropout [5] is not used. The learning rate starts from 0.1 and is divided by 10 when the number of iterations reaches 40K and 60K, respectively, and the total number of iterations is 80K. We use a standard weight decay of 0.0002. The nested structure is implemented in all layers for adaptive deep compression and in all residual blocks except the first convolutional and the last fully-connected layers for the rest of the applications based on the aforementioned architectures, where we learn different fully-connected weights in the final layer to address different purposes (e.g., different output dimensionality for hierarchical classification).

2.2. ImageNet dataset

NestedNet was constructed based on the ResNet-18 architecture following the instruction in [2] for the ImageNet

Table 1. Taxonomy of the ImageNet-Subtree dataset. (·) denotes the number of subclasses for each intermediate category.

Superclass	Intermediate class	Subclass
Natural object	Fruit (9)	Strawberry, Orange, Lemon, Fig, Pineapple, Banana, Jackfruit, Custard apple, Pomegranate
Plant	Vegetable (9)	Head cabbage, Broccoli, Cauliflower, Zucchini, Spaghetti squash, Acorn squash, Butternut squash, Cucumber, Artichoke
	Flower (3)	Daisy, Yellow lady's slipper, Cardoon
Animal	Dog (14)	Siberian husky, Australian terrier, English springer, Walker hound, Weimaraner, Soft coated wheaten terrier, Old English sheepdog, French bulldog, Basenji, Bernese mountain dog, Maltese dog, Doberman, Boston bull, Greater Swiss mountain dog
	Cat (5)	Egyptian cat, Persian cat, Tiger cat, Siamese cat, Madagascar cat
	Fish (10)	Great white shark, Tiger shark, Hammerhead, Electric ray, Stingray, Barracouta, Coho, Tench, Goldfish, Eel
	Bird (10)	Goldfinch, Robin, Bulbul, Jay, Bald eagle, Vulture, Peacock, Macaw, Hummingbird, Black swan
Artifact	Instrument (10)	Grand piano, Drum, Maraca, Cello, Violin, Harp, Acoustic guitar, Trombone, Harmonica, Sax
	Vehicle (10)	Airship, Speedboat, Yawl, Trimaran, Submarine, Mountain bike, Freight car, Passenger car, Minivan, Sports car
	Furniture (10)	Park bench, Barber chair, Throne, Folding chair, Rocking chair, Studio couch, Toilet seat, Desk, Pool table, Dining table
	Construction (10)	Suspension bridge, Viaduct, Barn, Greenhouse, Palace, Monastery, Library, Boathouse, Church, Mosque

dataset, where the numbers of channels in the core and medium level networks were set to quarter and half of the number of all channels, respectively, for every convolutional layer, as mentioned in the main paper. We set different fully-connected layers in the last output layer as performed in the CIFAR datasets. We use the SGD optimizer with momentum of 0.9 and the Nesterov method with the size of a mini-batch of 256 and the weight decay of 0.0001. The learning rate starts from 0.1 and divided by 10 when the number of epochs reaches 15K, 30K, and 45K, respectively, and the total number of epochs is 50K. For the dataset, we learn nested parameters sequentially from core to full level for every iteration instead of learning them simultaneously.

2.3. Consensus in NestedNet

For NestedNet-L described in Section 5.4 in the main paper, which incorporates multiple knowledge from all nested levels, we add a fully-connected layer, called a consensus layer, to the concatenated vector of all outputs in NestedNet, and the consensus layer again produces an output vector whose size is the number of classes. Note that the consensus layer is learned after NestedNet is trained in this work, but we can learn the whole network including the consensus layer simultaneously. When we address the hierarchical classification problem for the CIFAR-100 dataset [4] in Section 5.3 in the main paper, rather than just concatenating the two level outputs, we collect additional fine class output (whose dimensionality is 100) in the core level network, which requires another fully-connected layer in the final layer in NestedNet to produce an output of different dimensionality, and then learn the consensus layer using concatenation of the three outputs (two fine class outputs from both full and core levels and one coarse class output from the core level) for better prediction. We also average two fine class outputs from both level networks to build NestedNet-A for the hierarchical classification problem. For more accurate inference, one can append more layers with nonlinearity in the top of the network, while this practice only adds a layer without nonlinearity which may not achieve further performance gain for a certain problem. For ImageNet, we constructed two consensus variants of NestedNet in a similar way for CIFAR-100. When handling the knowledge distillation problem, we use the designed number of output features learned from NestedNet to construct NestedNet-A and NestedNet-L. We use the SGD optimizer without momentum for both knowledge distillation and hierarchical classification in learning NestedNet-L. To yield the best performance, the learning rate for all consensus layers starts from 0.1 and is divided by 10 when the number of iterations reaches 20K, 30K and 40K, respectively, and the total number of iterations is set to 50K.

3. Activation Feature Maps

Figure 2 shows activation feature maps, which are outputs from different layers in NestedNet when feeding train and test images of the CIFAR-10 dataset [4] to the learned network. Each row represents the maps obtained in each network with different nested level from core-level (top) to full-level (bottom). Note that the maps illustrated here are printed out using the filters in the current level network, which do not include the filters already computed in the sub-level networks, to see what filters in higher level networks learn (i.e., increments from the sub-level networks).

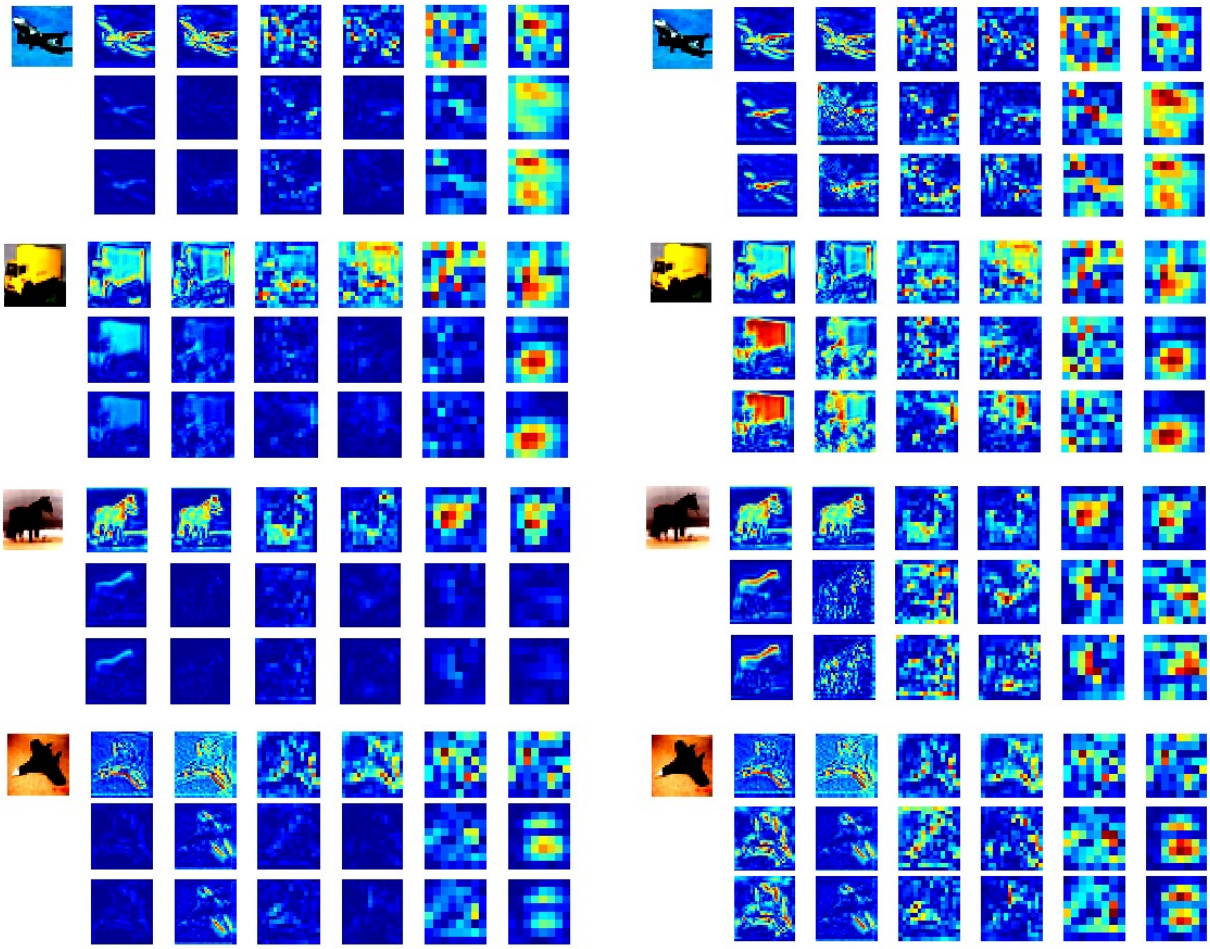


Figure 2. Activation maps in NestedNet according to different layers on train (first) and test (second to fourth) images for the CIFAR-10 dataset [4]. The maps scaled consistently in layer-wise (left) and the individual maps without keeping the consistent scale (right) are provided for better understanding. Each row for every image represents the images in each internal network, from core-level (top) to full-level (bottom). Best viewed in color.

We also provide additional activation maps for the same images that show the individual images without keeping the consistent scale when drawing the figures (right column in the figure), since the filters in higher level networks sometimes produce small values which are difficult to observe as shown in the left column of the figure. From the figure, we can see that the learned filters in the higher level networks also catch the important and complementary features, even though they are marginal compared to those in the core level network.

References

[1] X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, 2010. 1

[2] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 1

[3] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning (ICML)*, 2015. 1

[4] A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images. 2009. 1, 2, 3

[5] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research (JMLR)*, 15(1):1929–1958, 2014. 1

[6] S. Zagoruyko and N. Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016. 1