NestedNet: Learning Nested Sparse Structures in Deep Neural Networks

Eunwoo Kim   Chanho Ahn   Songhwai Oh

Department of ECE and ASRI, Seoul National University, South Korea

{kewoo15, mychahn, songhwai}@snu.ac.kr

Abstract

Recently, there have been increasing demands to construct compact deep architectures to remove unnecessary redundancy and to improve the inference speed. While many recent works focus on reducing the redundancy by eliminating unneeded weight parameters, it is not possible to apply a single deep network for multiple devices with different resources. When a new device or circumstantial condition requires a new deep architecture, it is necessary to construct and train a new network from scratch. In this work, we propose a novel deep learning framework, called a nested sparse network, which exploits an n-in-1-type nested structure in a neural network. A nested sparse network consists of multiple levels of networks with a different sparsity ratio associated with each level, and higher level networks share parameters with lower level networks to enable stable nested learning. The proposed framework realizes a resource-aware versatile architecture as the same network can meet diverse resource requirements, i.e., anytime property. Moreover, the proposed nested network can learn different forms of knowledge in its internal networks at different levels, enabling multiple tasks using a single network, such as coarse-to-fine hierarchical classification. In order to train the proposed nested network, we propose efficient weight connection learning and channel and layer scheduling strategies. We evaluate our network in multiple tasks, including adaptive deep compression, knowledge distillation, and learning class hierarchy, and demonstrate that nested sparse networks perform competitively, but more efficiently, compared to existing methods.

1. Introduction

Deep neural networks have recently become a standard architecture due to their significant performance improvement over the traditional machine learning models in a number of fields, such as image recognition [10, 13, 18], object detection [24], image generation [7], and natural language processing [4, 26]. The successful outcomes are derived from the availability of massive labeled data and compu-
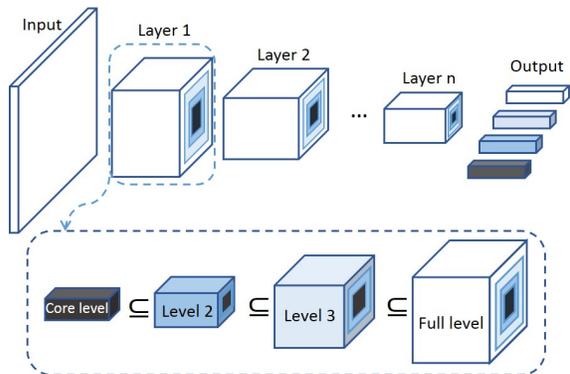

Figure 1.   Conceptual illustration of the nested sparse network with n nested levels (n-in-1 network, n=4 here). A nested sparse network consists of internal networks from core level (with the sparsest parameters) to full level (with all parameters) and an internal network share its parameters with higher level networks, making a network-in-network structure. Since the nested sparse network produces multiple different outputs, it can be leveraged for multiple tasks as well as multiple devices with different resources.

tational power to process such data. What is more, many studies have been conducted toward very deep and dense models [10, 13, 25, 32] to achieve further performance gain. Despite of the success, the remarkable progress is accomplished at the expense of intensive computational and memory requirements, which can limit deep networks for a practical use, especially on mobile devices with low computing capability. In particular, if the size of a network architecture is designed to be colossal, it may be problematic for the network to achieve mission-critical tasks on a commercial device which requires a real-time operation.

Fortunately, it is well-known that there exists much redundancy in most of deep architectures, i.e., a few number of network parameters represent the whole deep network in substance [20]. This motivates many researchers to exploit the redundancy from multiple points of view. The concept of sparse representation is to elucidate the redundancy by representing a network with a small number of representative parameters. Most of sparse deep networks prune connections with insignificant contributions [3, 8, 9, 23, 30, 34],

prune the number of channels [11,21,23], or prune the number of layers [28] by sparse regularization. Another regularization strategy to eliminate the network redundancy is low-rank approximation which approximates weight tensors by minimizing the reconstruction error between the original network and the reduced network [14,16,27,33,34]. Weight tensors can be approximated by decomposing into tensors of pre-specified sizes [14,16,27,33] or by solving a nuclear-norm regularized optimization problem [34].

Obviously, developing a compact deep architecture is beneficial to satisfy the specification of a device with low capacity. However, it is difficult for a learned compact network to be adjusted for different hardware specifications (e.g., different sparsity levels), since a deep neural network normally learns parameters for a given task. When a new model or a device with a different computing budget is required, we usually define a new network again manually by trial and error. Likewise, if a different form of knowledge is required in a trained network, it is hard to keep the learned knowledge while training using the same network again or using a new network [12]. In general, to perform multiple tasks, we need multiple networks at the cost of considerable computation and memory footprint.

In this work, we aim to exploit a nested structure in a deep neural architecture which realizes an n-in-1 versatile network to conduct multiple tasks within a single neural network (see Figure 1). In a nested structure, network parameters are assigned to multiple sets of nested levels, such that a low level set is a subset of parameters to a higher level set. Different sets can capture different forms of knowledge according to the type (or amount) of information, making it possible to perform multiple tasks using a single network.

To this end, we propose a nested sparse network, termed *NestedNet*, which consists of multiple levels of networks with different sparsity ratios (nested levels), where an internal network with lower nested level (higher sparsity) shares its parameters with other internal networks with higher nested levels in a network-in-network fashion. Thus, a lower level internal network can learn common knowledge while a higher level internal network can learn task-specific knowledge. It is well-known that early layers of deep neural networks share general knowledge and later layers learn task-specific knowledge. A nested network learns more systematic hierarchical representation and has an effect of grouping analogous filters for each nested level as shown in Section 5.1. NestedNet also enjoys another useful property, called anytime property [19, 35], and it can produce early (coarse) prediction with a low level network and more accurate (fine) answer with a higher level network. Furthermore, unlike existing networks, the nested sparse network can learn different forms of knowledge in its internal networks with different levels. Hence, the same network can be applied to multiple tasks satisfying different

resource requirements, which can reduce the efforts to train separate existing networks. In addition, consensus of different knowledge in a nested network can further improve the performance of the overall network. In order to exploit the nested structure, we present several pruning strategies which can be used to learn parameters from scratch using off-the-shelf deep learning libraries. We also provide applications, in which the nested structure can be applied, such as adaptive deep compression, knowledge distillation, and hierarchical classification. Experimental results demonstrate that NestedNet performs competitively compared to popular baseline and other existing sparse networks. In particular, our results in each application (and each data) are produced from a single nested network, making NestedNet highly efficient compared with currently available approaches.

In summary, the main contributions of this work are:

- We present an efficient connection pruning method, which learns sparse connections from scratch. We also provide channel and layer pruning by scheduling to exploit the nested structure to avoid the need to train multiple different networks.

- We propose an n-in-1 nested sparse network to realize the nested structure in a deep network. The nested structure enables not only resource-aware anytime prediction but knowledge-aware adaptive learning for various tasks which are not compatible with existing deep architectures. Besides, consensus of multiple knowledge can improve the prediction of NestedNet.

- The proposed nested networks are performed on various applications in order to demonstrate its efficiency and versatility at comparable performance.

2. Related Work

A naïve approach to compress a deep neural network is to prune network connections by sparse approximation. Han et al. [9] proposed an iterative prune and retrain approach using the l_1- or l_2-norm regularization. Zhou et al. [34] proposed a forward-backward splitting method to solve the l_1-norm regularized optimization problem. Note that the weight pruning methods with non-structured sparsity can be difficult to achieve valid speed-up using standard machines due to their irregular memory access [28]. Channel pruning approaches were proposed by structured sparsity regularization [28] and channel selection methods [11,23]. Since they reduce the actual number of parameters, they have benefits of computational and memory resources compared to the weight connection pruning methods. Layer pruning [28] is another viable approach for compression when the parameters associated with a layer has little contributions in a deep neural network using short-cut connection [10]. There is another line of compressing deep networks by low-rank

approximation, where weight tensors are approximated by low-rank tensor decomposition [14,16,27,33] or by solving a nuclear-norm regularized optimization problem [34]. It can save memory storage and enable valid speed-up when learning and inferencing the network. The low-rank approximation approaches, however, normally require a pre-trained model when optimizing parameters to reduce the reconstruction error with the original learned parameters.

It is important to note here that the learned networks using the above compression approaches are difficult to be utilized for different tasks, such as different compression ratios, since the learned parameters are trained for a single task (or a given compression ratio). If a new compression ratio is required, one can train a new network with manual model parameter tuning from scratch or further tune the trained network to suit the new demand[1], and this procedure will be conducted continually whenever the form of the model is changed, requiring additional resources and efforts. This difficulty can be fully addressed using the proposed nested sparse network. It can embody multiple internal networks within a network and perform different tasks at the cost of learning a single network. Furthermore, since the nested sparse network is constructed from scratch, the effort to learn a baseline network is not needed.

There have been studies to build a compact network from a learned large network, called knowledge distillation, while maintaining the knowledge of the large network [2, 12]. It shares intention with the deep compression approaches but utilizes the teacher-student paradigm to ease the training of networks [2]. Since it constructs a separate student network from a learned teacher network, its efficiency is also limited similar to deep compression models.

The proposed nested structure is also related to tree-structured deep architectures. Hierarchical structures in a deep neural network have been recently exploited for improved learning [15,19,29]. Yan et al. [29] proposed a hierarchical architecture that outputs coarse-to-fine predictions using different internal networks. Kim et al. [15] proposed a structured deep network that can enable model parallelization and a more compact model compared with previous hierarchical deep networks. However, since their networks do not have a nested structure since parameters in their networks form independent groups in the hierarchy, they cannot have the benefit of nested learning for sharing knowledge obtained from coarse- to fine-level sets of parameters. This limitation is discussed more in Section 5.3.

3. Compressing a Neural Network

Given a set of training examples $X = [x_1, x_2, ..., x_n]$ and labels $Y = [y_1, y_2, ..., y_n]$, where n is the number of

samples, a neural network learns a set of parameters $\mathcal{W}$ by minimizing the following optimization problem

$$\min_{\mathcal{W}} \mathcal{L}\Big(Y, f(X, \mathcal{W})\Big) + \lambda \mathcal{R}(\mathcal{W}), \qquad (1)$$

where $\mathcal{L}$ is a loss function between the network output and the ground-truth label, $\mathcal{R}$ is a regularizer which constrains weight parameters, and λ is a weighting factor balancing between loss and regularizer. $f(X, \mathcal{W})$ outputs according to the purpose of a task, such as classification (binary number) and regression (real number), through a chain of linear and nonlinear operations using the parameter $\mathcal{W}$. A set of parameters is represented by $\mathcal{W} = \{W_l\}_{1 \leq l \leq L}$, where L is the number of layers in a network, and $W_l \in \mathbb{R}^{k_w \times k_h \times c_i \times c_o}$ for a convolutional weight or $W_l \in \mathbb{R}^{c_i \times c_o}$ for a fully-connected weight in popular deep learning architectures such as AlexNet [18], VGG networks [25], and residual networks [10]. Here, k_w and k_h are the width and height of a convolutional kernel and c_i and c_o are the number of input and output channels (or activations[2]), respectively.

In order to exploit a sparse structure in a neural network, many studies usually try to enforce constraints on $\mathcal{W}$, such as sparsity using the l_1 [9, 34] or l_2 weight decay [9] and low-rank-ness using the nuclear-norm [34] or tensor factorization [14, 31]. However, many previous studies utilize a pre-trained network and then prune connections in the network to develop a parsimonious network, which usually requires significant additional computation.

4. Nested Sparse Networks

4.1. Sparsity learning by pruning

We investigate three pruning approaches for sparse deep learning: (entry-wise) weight connection pruning, channel pruning, and layer pruning, which are used for nested sparse networks described in Section 4.2.

To achieve weight connection pruning, pruning strategies were proposed to reduce learned parameters using a pre-defined threshold [9] and using a subgradient method [34]. However, they require additional pruning steps to sparsify a learned dense network. As an alternative, we propose an efficient sparse connection learning approach which learns from scratch without additional pruning steps using the standard optimization tool. The problem formulation can be constructed as follows:

$$\min_{\mathcal{W}} \mathcal{L}\Big(Y, f(X, \mathcal{P}_{\Omega_{\mathcal{M}}}(\mathcal{W}))\Big) + \lambda \mathcal{R}\Big(\mathcal{P}_{\Omega_{\mathcal{M}}}(\mathcal{W})\Big)$$
$$\text{s.t. } \mathcal{M} \triangleq \sigma(\alpha(\mathcal{W}) - \tau), \qquad (2)$$

where $\mathcal{P}(\cdot)$ is the projection operator and $\Omega_{\mathcal{M}}$ denotes the support set of $\mathcal{M} = \{M_l\}_{1 \leq l \leq L}$. $\alpha(\cdot)$ is the element-wise

---

[1]Additional tuning on a trained network with a new requirement can be accompanied by forgetting the learned knowledge [22].

[2]Activations denote neurons in fully-connected layers.

absolute operator and $\sigma(\cdot)$ is an activation function to encode binary output (such as the unit-step function) and τ is a pre-defined threshold value for pruning. Since the unit-step function makes learning the problem (2) by standard back-propagation difficult due to its discontinuity, we present a simple approximated unit-step function:

$$\sigma(x) = tanh(\gamma \cdot x_+), \qquad (3)$$

where $x_+ = max(x, 0)$, $tanh(\cdot)$ is the hyperbolic tangent function, and γ is a large value to mimic the slope of the unit-step function.[3] Note that $\mathcal{M}$ acts as an implicit mask of $\mathcal{W}$ to reveal sparse weight tensor. Once an element of $\mathcal{M}$ becomes 0, its corresponding weight is no longer updated in the optimization procedure, making no more contribution to the network. By solving (2), we construct a sparse deep network based on off-the-shelf deep learning libraries without additional efforts.

To achieve channel or layer pruning[4], we consider the following weight (or channel) scheduling problem:

$$\min_{\mathcal{W}} \ \mathcal{L}\Big(Y, f(X, \mathcal{P}_{\Omega_{\mathcal{M}'}}(\mathcal{W}))\Big) + \lambda \mathcal{R}\Big(\mathcal{P}_{\Omega_{\mathcal{M}'}}(\mathcal{W})\Big), \quad (4)$$

where $\mathcal{M}' = \{M_l'\}_{1 \le l \le L} \subseteq \mathcal{M}$ consists of binary weight tensors whose numbers of input and output channels (or activations for fully-connected layers) are reduced to smaller numbers than the numbers of channels (activations) in the baseline architecture to fulfill the demanded sparsity. In other words, we model a network with a single number of scheduled channels using $\mathcal{M}'$ and then optimize $\mathcal{W}$ in the network from scratch. Achieving multiple sparse networks by scheduling multiple numbers of channels is described in the following section. Similar to the channel pruning, implementing layer pruning is straight-forward by reducing the number of layers in repeated blocks [10]. In addition, pruning approaches can be combined for various nested structures as described in the next section.

4.2. Nested sparse networks

The goal of a nested sparse network is to represent an n-in-1 nested structure of parameters in a deep neural network to allow n nested internal networks as shown in Figure 1. In a nested structure, an internal network with lower (resp., higher) nested level gives higher (resp., lower) sparsity on parameters, where higher sparsity means a smaller number of non-zero entries. In addition, the internal network of the core level (resp., the lowest level) defines the most compact sub network among the internal networks and the internal network of the full level (resp., the highest level) defines

---

[3]We set $\gamma = 10^5$ and it is not sensitive to initial values of parameters when applying the popular initialization method [6] from our empirical experiences.

[4]Layer pruning can be applicable to the structure in which weights of the same size are repeated, such as residual networks [10].
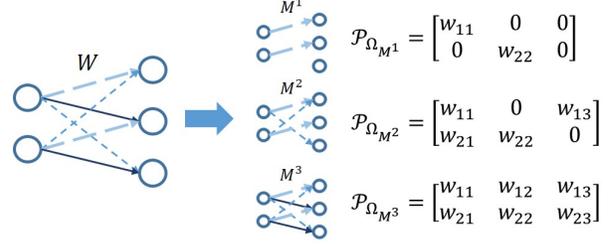

Figure 2. A graphical representation of nested parameters W by masks with three levels between fully-connected layers.

the fully dense network. Between them, there can be other internal networks with intermediate sparsity ratios. Importantly, an internal network of a lower nested level shares its parameters with other internal networks of higher nested levels.

Given a set of masks $\mathcal{M}$, a nested sparse network, where network parameters are assigned to multiple sets of nested levels, can be learned by optimizing the following problem:

$$\min_{\mathcal{W}} \frac{1}{l_n}\left(\sum_{j=1}^{l_n} \mathcal{L}\Big(Y, f(X, \mathcal{P}_{\Omega_{\mathcal{M}^j}}(\mathcal{W}))\Big)\right) + \lambda \mathcal{R}\Big(\mathcal{P}_{\Omega_{\mathcal{M}^{l_n}}}(\mathcal{W})\Big)$$
$$\text{s.t. } \mathcal{M}^j \subseteq \mathcal{M}^k, \ j \le k, \ \forall j, k \in [1, ..., l_n], \qquad (5)$$

where l_n is the number of nested levels. Since a set of masks $\mathcal{M}^j = \{M_l^j\}_{1 \le l \le L}$ represents the set of j-th nested level weights by its binary values, $\mathcal{P}_{\Omega_{\mathcal{M}^j}}(\mathcal{W}) \subseteq \mathcal{P}_{\Omega_{\mathcal{M}^k}}(\mathcal{W}), j \le k, \forall j, k \in [1, ..., l_n]$. A simple graphical illustration of nested parameters between fully-connected layers is shown in Figure 2. By optimizing (5), we can build a nested sparse network with nested levels by $\mathcal{M}$.

In order to find a set of masks $\mathcal{M}$, we apply three pruning approaches described in Section 4.1. First, for realizing a nested structure in entry-wise weight connections, masks are estimated by solving the weight connection pruning problem (2) with l_n different thresholds iteratively. Specifically, once the mask $\mathcal{M}^k$ consisting of the k-th nested level weights is obtained in a network[5], we further train the network from the masked weight $\mathcal{P}_{\Omega_{\mathcal{M}^k}}(\mathcal{W})$ using a higher value of threshold to get another mask giving a higher sparsity, and this procedure is conducted iteratively until reaching the sparsest mask of the core level. This strategy is helpful to find sparse dominant weights [9], and our network trained using this strategy performs better than a network whose sparse mask is obtained randomly and other sparse networks in Section 5.1.

For a nested sparse structure in convolutional channels or layers, we schedule a set of masks $\mathcal{M}$ according to the type

---

[5]Since we use the approximation in (3), an actual binary mask is obtained by additional thresholding after the mask is estimated.

of pruning. In the channel-wise scheduling, the number of channels in convolutional layers and the dimensions in fully-connected layers are scheduled to pre-specified numbers for all scheduled layers. The scheduled weights are learned by solving (5) without performing the mask estimation phase in (2). Mathematically, we represent weights from the first (core) level weight $W_l^1 \in \mathbb{R}^{k_w \times k_h \times i_1 \times o_1}$ to the full level weight $W_l = W_l^{l_n} \in \mathbb{R}^{k_w \times k_h \times c_i \times c_o}$, where $c_i = \sum_m i_m$ and $c_o = \sum_m o_m$, between l-th and $(l+1)$-th convolutional layers as

$$W_l^1 = W_l^{1,1},$$

$$W_l^2 = \begin{bmatrix} W_l^{1,1} & W_l^{1,2} \\ W_l^{2,1} & W_l^{2,2} \end{bmatrix},$$

$$\vdots$$

$$W_l = W_l^{l_n} = \begin{bmatrix} W_l^{1,1} & W_l^{1,2} & \cdots & W_l^{1,l_n} \\ W_l^{2,1} & W_l^{2,2} & \cdots & W_l^{2,l_n} \\ \vdots & \vdots & \vdots & \vdots \\ W_l^{l_n,1} & W_l^{l_n,2} & \cdots & W_l^{l_n,l_n} \end{bmatrix}. \tag{6}$$

Figure 3 illustrates the nested sparse network with channel scheduling, where different color represents weights in different nested level except its shared sub-level weights. For the first input layer, observation data is not scheduled in this work (i.e., c_i is not divided). Unlike the nested sparse network with the weight pruning method, which holds a whole-size network structure for any nested levels, channel scheduling only keeps and learns the parameters corresponding to the number of scheduled channels associated with a nested level, making valid speed-up especially for an inference phase.

Likewise, we can schedule the number of layers and its corresponding weights in a repeated network block and learn parameters by solving (5). Note that for a residual network which consists of $6n_b + 2$ layers [10], where n_b is the number of residual blocks, if we schedule $n_b = \{2,3,5\}$, our single nested residual network with $l_n = 3$ consists of three residual networks of size 14, 20, and 32 in the end. Among them, the full level network with $n_b = 5$ has the same number of parameters to the conventional residual network of size 32 without introducing further parameters.

4.3. Applications

Adaptive deep compression. Since a nested sparse network is constructed under the weight connection learning, it can apply to deep compression [8]. Furthermore, the nested sparse network realizes adaptive deep compression because of its anytime property [35] which makes it possible to provide various sparse networks and infer adaptively using a learned internal network with a sparsity level suitable for the required specification. For this problem, we apply weight pruning and channel scheduling presented in Section 4.2.
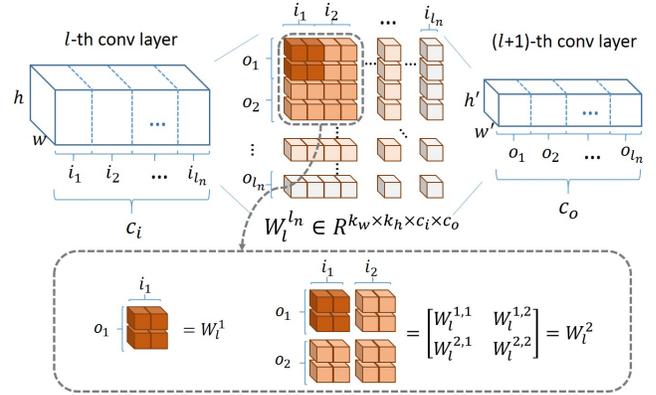

Figure 3. A graphical representation of channel scheduling using the nested parameters between l-th and $(l+1)$-th convolutional layers in a nested sparse network where both layers are scheduled. i_k and o_k denote additional numbers of input and output channels in the k-th level, respectively.

Knowledge distillation. Knowledge distillation is used to represent knowledge compactly in a network [12]. Here, we apply channel and layer scheduling approaches to make small-size sub-networks as shown in Figure 4. We train all internal networks, one full-level and $l_n - 1$ sub-level networks, simultaneously from scratch without pre-training the full-level network. Note that the nested structure in sub-level networks may not be necessarily coincided with the combination of channel and layer scheduling (e.g., subset constraint is not satisfied for Figure 4 (b) and (c)) according to a design choice.

Hierarchical classification. In a hierarchical classification problem [29], a hierarchy can be modeled as an internal network with a nested level. For example, we model a nested network with two nested levels for the CIFAR-100 dataset [17] as it has 20 super classes, where each class has 5 subclasses (a total 100 subclasses). It enables nested learning to perform coarse to fine representation and inference. We apply the channel pruning method since it can handle different output dimensionality.

5. Experiments

We have evaluated NestedNet based on popular deep architectures, ResNet-n [10] and WRN-n-k [32], where n is the number of layers and k is the scale factor on the number of convolutional channels for the above three applications. Since it is difficult to compare fairly with other baseline and sparse networks due to their non-nested structure, we provide a one-to-one comparison between internal networks in NestedNet and their corresponding independent baselines or other published networks of the same network structure. NestedNet was performed on three benchmark datasets: CIFAR-10, CIFAR-100 [17], and ImageNet [18].
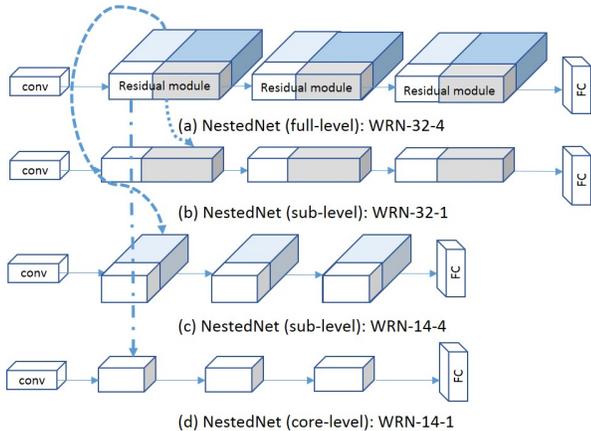
Figure 4. NestedNet with four nested levels using channel and layer scheduling for knowledge distillation.

The test time is computed for a batch set of the same size to the training phase. All NestedNet variants and other compared baselines were implemented using the TensorFlow library [1] and processed by an NVIDIA TITAN X graphics card. Implementation details of our models are described in the supplementary material.

5.1. Adaptive deep compression

We applied the weight connection and channel pruning approaches described in Section 4.1 based on ResNet-56 to compare with the state-of-the-art network (weight connection) pruning [9] and channel pruning approaches [11, 21]. We implemented the iterative network pruning method for [9] under our experimental environment giving the same baseline accuracy, and results of channel pruning approaches [11, 21] under the same baseline network were refereed from [11]. To compare with the channel pruning approaches, our nested network was constructed with two nested levels, full-level (1× compression) and core-level (2× compression), and to compare with the network pruning method, we constructed another NestedNet with three internal networks (1×, 2×, and 3× compressions) by setting $\boldsymbol{\tau} = (\tau_{1\times}, \tau_{2\times}, \tau_{3\times}) = (0, 0.015, 0.025)$, where the full-level networks give the same result to the baseline network, ResNet-56 [10]. In the experiment, we also provide results of the three-level nested network (1×, 2×, and 3× compressions), which is learned using random sparse masks, in order to verify the effectiveness of the proposed weight connection learning method in Section 4.1.

Table 1 shows the classification accuracy of the compared networks for the CIFAR-10 dataset. For channel pruning, NestedNet gives smaller performance loss from baseline than recently proposed channel pruning approaches [11, 21] under the same reduced parameters (2×), even though the baseline performance is not the same due to their different implementation strategies. For weight con-

Table 1. Deep compression results using ResNet-56 for the CIFAR-10 dataset. (·) denotes the compression rate of parameters from the baseline network. Baseline results are obtained by author's implementation without pruning (1×).

Method	Accuracy	Baseline
Filter pruning [21] (2×)	91.5%	92.8%
Channel pruning [11] (2×)	91.8%	92.8%
NestedNet - channel pruning (2×)	92.9%	93.4%
Network pruning [9] (2×)	93.4%	93.4%
NestedNet - random mask (2×)	91.2%	93.4%
NestedNet - weight pruning (2×)	93.4%	93.4%
Network pruning [9] (3×)	92.6%	93.4%
NestedNet - random mask (3×)	85.1%	93.4%
NestedNet - weight pruning (3×)	92.8%	93.4%

nection pruning, ours performs better than network pruning [9] on average. They show no accuracy compromise under 2× compression, but ours gives better accuracy than [9] under 3× compression. Here, the weight connection pruning approaches outperform channel pruning approaches including our channel scheduling based network under 2× compression, since they prune unimportant connections in element-wise while channel pruning approaches eliminate connections in group-wise (thus dimensionality itself is reduced) which can produce information loss. Note that the random connection pruning gives the poor performance, confirming the benefit of the proposed connection learning approach in learning the nested structure.

Figure 5(a) represents learned filters (brighter represents more intense) of the nested network with the channel pruning approach using ResNet-56 with three levels (1×, 2×, and 4× compressions) where the size of the first convolutional filters were set to 7 × 7 to see observable large size filters under the same performance. As shown in the figure, the connections in the core-level internal network are dominant and upper-level filters, which do not include their sub-level filters (when drawing the filters), have lower importance than core-level filters which may learn side information of the dataset. We also provide quantitative results for filters in three levels using averaged normalized mutual information for all levels: 0.38 ± 0.09 for within-level and 0.08 ± 0.03 for between-level, which reveal that the nested network learns more diverse filters between nested levels than within levels and it has an effect of grouping analogous filters for each level. Figure 5(b) shows the activation maps (layer outputs) of an image for different layers. For more information, we provide additional activation maps for both train and test images in the supplementary material.

5.2. Knowledge distillation

To show the effectiveness of nested structures, we evaluated NestedNets using channel and layer scheduling for knowledge distillation where we learned all internal net-
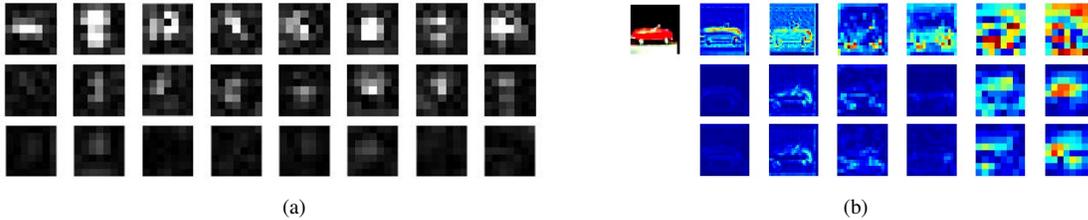
(a)   (b)

Figure 5. Results from NestedNet on the CIFAR-10 dataset. (a) Learned filters from the first convolutional layer. (b) Activation feature maps from a train image (upper left). Each column represents a different layer (the layer number increases from left to right). Each row represents the images in each internal network, from core-level (top) to full-level (bottom). Best viewed in color.

works jointly rather than learning a distilled network from a pre-trained model in the literature [12]. The proposed network was constructed under the WRN architecture [32] (here WRN-32-4 was used). We set the full-level network to WRN-32-4 and applied (1) channel scheduling with $\frac{1}{4}$ scale factor (WRN-32-1 or ResNet-32), (2) layer scheduling with $\frac{2}{5}$ scale factor (WRN-14-4), and (3) combined scheduling for both channel and layer (WRN-14-1 or ResNet-14). In the scenario, we did not apply the nested structure for the first convolutional layer and the final output layer. We applied the proposed network to CIFAR-10 and CIFAR-100.

Figure 6 shows the comparison between NestedNet with four internal networks, and their corresponding baseline networks learned independently for the CIFAR-10 dataset. We also provide test time of every internal network.[6] As observed in the figure, NestedNet performs competitively compared to its baseline networks for most of the density ratios. Even though the total number of parameters to construct the nested sparse network is smaller than that to learn its independent baseline networks, the shared knowledge among the multiple internal networks can compensate for the handicap and give the competitive performance to the baselines. When it comes to test time, we achieve valid speed-up for the internal networks with reduced parameters, from about $1.5\times$ (37% density) to $8.3\times$ speed-up (2.3% density). Table 2 shows the performance of NestedNet, under the same baseline structure to the previous example, for the CIFAR-100 dataset. N_C and N_P denote the number of classes and parameters, respectively. In the problem, NestedNet is still comparable to its corresponding baseline networks on average, which requires similar resource to the single baseline of full level.

5.3. Hierarchical classification

We evaluated the nested sparse network for hierarchical classification. We first constructed a two-level nested network for the CIFAR-100 dataset, which consists of two-level hierarchy of classes, and channel scheduling was applied to handle different dimensionality in the hierarchical
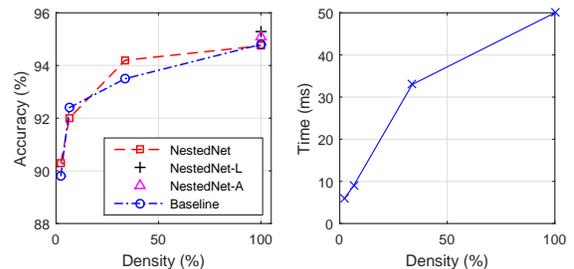

Figure 6. Knowledge distillation results w.r.t accuracy (left) and time (right) using NestedNet for the CIFAR-10 dataset.

structure of the dataset. We compared with the state-of-the-art architecture, SplitNet [15], which can address class hierarchy. Following the practice in [15], NestedNet was constructed under WRN-14-8 and we adopted WRN-14-4 as a core internal network ($4\times$ compression). Since the number of parameters in SplitNet is reduced to nearly 68% from the baseline, we constructed another NestedNet based on the WRN-32-4 architecture which has the almost same number of parameters as SplitNet.

Table 3 shows the performance comparison among the compared networks. Overall, our two NestedNets based on different architectures give the better performance than their baselines for all cases, since ours can learn rich knowledge from not only learning the specific classes but learning their abstract level (super-class) knowledge within the nested network, compared to merely learning independent class hierarchy. NestedNet also outperforms SplitNet for both architectures. While SplitNet learns parameters which are divided into independent sets, NestedNet learns shared knowledge for different tasks which can further improve the performance by its combined knowledge obtained from multiple internal networks. The experiment shows that the nested structure can realize encompassing multiple semantic knowledge in a single network to accelerate learning. Note that if the number of internal networks increases for more hierarchy, the amount of resources saved increases.

We also provide experimental results on the ImageNet (ILSVRC 2012) dataset [5]. From the dataset, we collected a subset, which consists of 100 diverse classes including

---

[6]Since the baseline networks require the same number of parameters and time as our networks, we just present test time of our networks.

Table 2. Knowledge distillation results on the CIFAR-100 dataset. ⋆ indicates that the approximated total resource for the nested network.

Network	Architecture	N_P	Density	Memory	Accuracy	Baseline	Test time	Consensus ($N_C = 100$)
NestedNet	WRN-32-4	7.4M⋆	100%	28.2MB⋆	75.5%	75.7%	52ms (1×)	
	WRN-14-4	2.7M	37%	10.3MB	74.3%	73.8%	35ms (1.5×)	NestedNet-A: 77.4%
	WRN-32-1	0.47M	6.4%	1.8MB	67.1%	67.5%	10ms (5.2×)	NestedNet-L: 78.1%
	WRN-14-1	0.18M	2.4%	0.7MB	64.3%	64.0%	7ms (7.4×)	

Table 3. Hierarchical classification results on the CIFAR-100 dataset. ⋆ indicates that the approximated total resource for the nested network.

Network	Architecture	N_C	N_P	Accuracy
Baseline	WRN-14-4	20	2.7M	82.4%
	WRN-14-8	100	10.8M	75.8%
NestedNet	WRN-14-4	20	2.7M	83.9%
	WRN-14-8	100	10.8M⋆	77.3%
NestedNet-A	WRN-14-8	100	10.9M⋆	78.3%
NestedNet-L	WRN-14-8	100	10.9M⋆	78.0%
SplitNet [15]	WRN-14-8	100	7.4M	74.9%
Baseline	WRN-32-2	20	1.8M	82.1%
	WRN-32-4	100	7.4M	75.7%
NestedNet	WRN-32-2	20	1.8M	83.7%
	WRN-32-4	100	7.4M⋆	76.6%
NestedNet-A	WRN-32-4	100	7.4M⋆	78.0%
NestedNet-L	WRN-32-4	100	7.4M⋆	77.7%

Table 4. Hierarchical classification results on ImageNet.

Network	N_C	N_P	Accuracy
Baseline	4	0.7M	92.8%
	11	2.8M	89.2%
	100	11.1M	79.8%
NestedNet	4	0.7M	94.0%
	11	2.8M	90.2%
	100	11.1M	79.9%
NestedNet-A	100	11.1M	80.2%
NestedNet-L	100	11.1M	80.3%

level network for hierarchical classification. See the supplementary material for more details. While the overhead of combining outputs of different levels of NestedNet is negligible, as shown in the results for knowledge distillation and hierarchical classification, the two consensus approaches outperform the existing structures including NestedNet of full-level under the similar number of parameters. Notably, NestedNet of full-level in hierarchical classification gives better performance than that in knowledge distillation under the same architecture, WRN-32-4, since it has rich knowledge by incorporating coarse class information in its architecture without introducing additional structures.

6. Conclusion

We have proposed a nested sparse network, named NestedNet, to realize an n-in-1 nested structure in a neural network, where several networks with different sparsity ratios are contained in a single network and learned simultaneously. To exploit such structure, novel weight pruning and scheduling strategies have been presented. NestedNet is an efficient architecture to incorporate multiple knowledge or additional information within a neural network, while existing networks are difficult to embody such structure. NestedNets have been extensively tested on various applications and demonstrated that it performs competitively, but more efficiently, compared to existing deep architectures.

natural objects, plants, animals, and artifacts. We constructed a three-level hierarchy and the numbers of super and intermediate classes are 4 and 11, respectively (a total 100 subclasses). Taxonomy of the dataset is summarized in the supplementary material. The number of train and test images are 128,768 and 5,000, respectively, which were collected from the original ImageNet dataset [5]. NestedNet was constructed based on the ResNet-18 architecture following the instruction in [10] for the ImageNet dataset, where the numbers of channels in the core and intermediate level networks were set to quarter and half of the number of all channels, respectively, for every convolutional layer. Table 4 summarizes the hierarchical classification results for the ImageNet dataset. The table shows that NestedNet, whose internal networks are learned simultaneously in a single network, outperforms its corresponding baseline networks for all nested levels.

5.4. Consensus of multiple knowledge

One important benefit of NestedNet is to leverage multiple knowledge of internal networks in a nested structure. To utilize the benefit, we appended another layer at the end, which we call a consensus layer, to combine outputs from all nested levels for more accurate prediction by 1) averaging (*NestedNet-A*) or 2) learning (*NestedNet-L*). For NestedNet-L, we simply added a fully-connected layer to the concatenated vector of all outputs in NestedNet, where we additionally collected the fine class output in the core

References

[1] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, et al. TensorFlow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*, 2016. 6

[2] R. Adriana, B. Nicolas, K. S. Ebrahimi, C. Antoine, G. Carlo, and B. Yoshua. FitNets: Hints for thin deep nets. In *International Conference on Learning Representations (ICLR)*, 2015. 3

[3] J. M. Alvarez and M. Salzmann. Learning the number of neurons in deep networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2016. 1

[4] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin. A neural probabilistic language model. *Journal of Machine Learning Research (JMLR)*, 3(Feb):1137–1155, 2003. 1

[5] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (CVPR)*, 2009. 7, 8

[6] 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. 4

[7] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems (NIPS)*, 2014. 1

[8] S. Han, H. Mao, and W. J. Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. In *International Conference on Learning Representations (ICLR)*, 2016. 1, 5

[9] S. Han, J. Pool, J. Tran, and W. Dally. Learning both weights and connections for efficient neural network. In *Advances in Neural Information Processing Systems (NIPS)*, 2015. 1, 2, 3, 4, 6

[10] 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, 2, 3, 4, 5, 6, 8

[11] Y. He, X. Zhang, and J. Sun. Channel pruning for accelerating very deep neural networks. In *International Conference on Computer Vision (ICCV)*, 2017. 2, 6

[12] G. Hinton, O. Vinyals, and J. Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015. 2, 3, 5, 7

[13] G. Huang, Z. Liu, K. Q. Weinberger, and L. van der Maaten. Densely connected convolutional networks. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 1

[14] M. Jaderberg, A. Vedaldi, and A. Zisserman. Speeding up convolutional neural networks with low rank expansions. *arXiv preprint arXiv:1405.3866*, 2014. 2, 3

[15] J. Kim, Y. Park, G. Kim, and S. J. Hwang. SplitNet: Learning to semantically split deep networks for parameter reduction and model parallelization. In *International Conference on Machine Learning (ICML)*, 2017. 3, 7, 8

[16] Y.-D. Kim, E. Park, S. Yoo, T. Choi, L. Yang, and D. Shin. Compression of deep convolutional neural networks for fast and low power mobile applications. In *International Conference on Learning Representations (ICLR)*, 2016. 2, 3

[17] A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images. 2009. 5

[18] A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2012. 1, 3, 5

[19] G. Larsson, M. Maire, and G. Shakhnarovich. FractalNet: Ultra-deep neural networks without residuals. *arXiv preprint arXiv:1605.07648*, 2016. 2, 3

[20] Y. LeCun, J. S. Denker, and S. A. Solla. Optimal brain damage. In *Advances in Neural Information Processing Systems (NIPS)*, 1990. 1

[21] H. Li, A. Kadav, I. Durdanovic, H. Samet, and H. P. Graf. Pruning filters for efficient convnets. In *International Conference on Learning Representations (ICLR)*, 2017. 2, 6

[22] Z. Li and D. Hoiem. Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017. 3

[23] Z. Liu, J. Li, Z. Shen, G. Huang, S. Yan, and C. Zhang. Learning efficient convolutional networks through network slimming. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2755–2763. IEEE, 2017. 1, 2

[24] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2015. 1

[25] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations (ICLR)*, 2015. 1, 3

[26] I. Sutskever, O. Vinyals, and Q. V. Le. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2014. 1

[27] C. Tai, T. Xiao, Y. Zhang, X. Wang, et al. Convolutional neural networks with low-rank regularization. *arXiv preprint arXiv:1511.06067*, 2015. 2, 3

[28] W. Wen, C. Wu, Y. Wang, Y. Chen, and H. Li. Learning structured sparsity in deep neural networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2016. 2

[29] Z. Yan, H. Zhang, R. Piramuthu, V. Jagadeesh, D. DeCoste, W. Di, and Y. Yu. HD-CNN: hierarchical deep convolutional neural networks for large scale visual recognition. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2015. 3, 5

[30] T.-J. Yang, Y.-H. Chen, and V. Sze. Designing energy-efficient convolutional neural networks using energy-aware pruning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 1

[31] X. Yu, T. Liu, X. Wang, and D. Tao. On compressing deep models by low rank and sparse decomposition. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 3

[32] S. Zagoruyko and N. Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016. 1, 5, 7

[33] X. Zhang, J. Zou, K. He, and J. Sun. Accelerating very deep convolutional networks for classification and detection. *IEEE transactions on pattern analysis and machine intelligence (TPAMI)*, 38(10):1943–1955, 2016. 2, 3

[34] H. Zhou, J. M. Alvarez, and F. Porikli. Less is more: Towards compact CNNs. In *European Conference on Computer Vision (ECCV)*. Springer, 2016. 1, 2, 3

[35] S. Zilberstein. Using anytime algorithms in intelligent systems. *AI magazine*, 17(3):73, 1996. 2, 5