Sequentially Aggregated Convolutional Networks

Yiwen Huang*†[1] Pinglai Ou*[2] Rihui Wu*[3] Ziyong Feng[4]
[1]Wenhua College, Huazhong University of Science and Technology [2]Virginia Tech
[3]University of Sydney [4]DeepGlint Technology Limited
nickgray0@gmail.com, kice@vt.edu, wrhkurt@gmail.com, ziyongfeng@deepglint.com

Abstract

Modern deep networks generally implement a certain form of shortcut connections to alleviate optimization difficulties. However, we observe that such network topology alters the nature of deep networks. In many ways, these networks behave similarly to aggregated wide networks. We thus exploit the aggregation nature of shortcut connections at a finer architectural level and place them within wide convolutional layers. We end up with a sequentially aggregated convolutional (SeqConv) layer that combines the benefits of both wide and deep representations by aggregating features of various depths in sequence. The proposed SeqConv serves as a drop-in replacement of regular wide convolutional layers and thus could be handily integrated into any backbone network. We apply SeqConv to widely adopted backbones including ResNet and ResNeXt, and conduct experiments for image classification on public benchmark datasets. Our ResNet based network with a model size of ResNet-50 easily surpasses the performance of the 2.35× larger ResNet-152, while our ResNeXt based model sets a new state-of-the-art accuracy on ImageNet classification for networks with similar model complexity. The code and pre-trained models of our work are publicly available at https://github.com/GroupOfAlchemists/SeqConv.

1. Introduction

Convolutional neural networks (CNNs) have gained overwhelming success for visual recognition owing to their representational ability. Recent work has shown that the depth of representation is of crucial importance to the performance of CNNs [26, 34, 10, 14]. Eldan *et al*. conclude that depth is a determinant factor of the expressiveness of neural networks [5]. Several studies [24, 40, 1] have also been conducted to investigate the width of the representation, it however does not seem to be the major concern of recent network architecture designs. The possibility to utilize the representational power of both wide and deep representations under a given model complexity has remained an unexplored problem.

Increasing depth by simply stacking more layers leads to optimization difficulties as the information flow gets gradually obscured by each layer during propagation [2, 30] in deep networks. An intuitive approach to ameliorate this problem is introducing shortcut connections towards farther layers to enable direct access to the guiding signal through propagation. This method has been shown particularly effective by various recently proposed state-of-the-art networks [10, 19, 14, 3], with its effectiveness further confirmed by visualizing the loss landscape of such networks [22].

Despite the fact that shortcut connections make it viable to optimize extremely deep networks as they help preserve the information flow, they seem to change the expected behaviors of deep networks. In many ways, these networks exhibit the property of having weak dependencies between consecutive layers while layers generally share strong dependencies in regular deep networks [36]. It is reported that ResNets [10], a typical network architecture with heavy use of shortcut connections, behave similarly to ensembles of many shallow networks [36], suggesting that such networks could be viewed from the aspect of a collection of several mostly independent subsections. This quality of having independent subsections and the aggregation nature of skip connections provide us the insight to link networks with shortcut connections to pseudo-wide networks.

The benefit of having wide representations lies in the fact that it allows for a larger feature space by introducing higher feature throughput to the network, however we argue that wide convolutional layers are not the only means to achieve this goal. A wide representation could also be collectively formulated by aggregating many transformations with small kernels. The shortcut connections are a case of aggregated transformations as they aggregate outputs from many layers, thus there is no surprise when we observe certain properties that resemble the behavior of a wide network on a

* Equal contribution.
† This work was done when Yiwen Huang was an intern at DeepGlint Technology Limited.

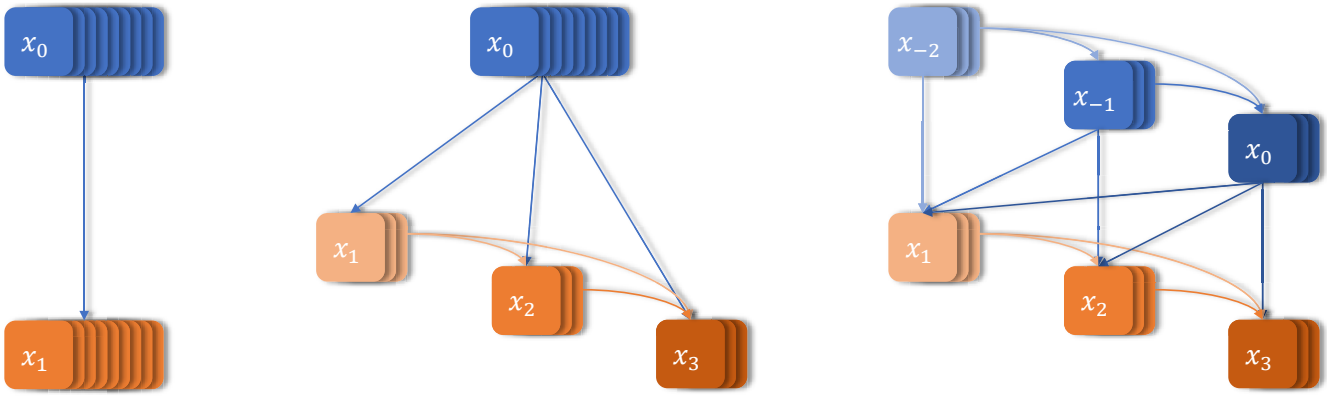

Figure 1. A regular wide convolutional layer (left), a sequentially aggregated convolutional layer (middle) and its windowed variant (right) with g=3.

deep and thin network with shortcut connections.

We propose a novel aggregation-based convolutional layer (SeqConv) to construct networks with the benefits of both wide and deep representations following the aggregation nature of shortcut connections. We divide a regular wide convolutional layer into several groups and place in-layer shortcut connections between each group. We then aggregate the outputs from all groups in sequence to formulate a collective wide representation. The SeqConv layer not only preserves the width of a regular convolutional layer, but as well introduces a hierarchical multi-path micro-architecture that is capable of representing heterogeneous kernels [27]. The representation capability of the layer is thus greatly enhanced, it is possible for a single SeqConv layer to produce multi-scale representation [34, 33] with deep hierarchical features. Our contributions in this paper are threefold:

- We propose sequentially aggregated convolutional (SeqConv) layers, along with several enhanced variants, that are capable of producing stronger representations than standard convolutional layers.
- We analyze the relations of SeqConv to DenseNet [14], and reinterpret the success of DenseNet with small growth rate from the perspective of sequentially aggregated wide representations. A windowed aggregation mechanism is also proposed to address the parameter redundancy and high computational cost of dense aggregation.
- We adopt SeqConv as the drop-in replacement of regular wide convolutional layers to construct networks for image classification. Our models achieve higher accuracy than significantly larger state-of-the-art models.

2. Related Work

Skip connectivity. Deep networks have been shown hard to optimize via gradient-based methods due to obstructed information flow through propagation, namely the diminishing feature reuse problem for forward propagation [30] and the vanishing gradient problem for backward propagation [2]. Networks with shortcut connections [10, 19, 14, 3] were proposed to alleviate such optimization difficulties by introducing shorter paths to farther layers and thus preserving the information flow through propagation. Several implementations of skip connectivity have been proposed to demonstrate the effectiveness of this network topology. Highway networks [30, 31] and residual networks [10, 11] construct skip connections with addition. Fractal networks [19] replace addition with element-wise mean which makes no distinction between signals from each path and thus allows a new form of regularization, Drop-path, to be applied. DenseNets [14] and its successors [12, 42] adopt concatenation to implement shortcut connections and attain favorable performance over previous work.

Ensembles of relatively shallow networks. Several analyses were conducted to investigate properties of the particularly effective residual networks. Veit *et al*. [36] reported that removing building blocks from residual networks or only keeping the shortcut paths did not lead to apparent accuracy drop. Huang *et al*. [15] randomly dropped residual blocks while training and actually obtained improved performance. Both studies suggest that layers in residual networks do not share strong dependencies between each other and such observation is not expected for a regular deep network. As reported by Veit *et al*. [36], removing layers from a VGG network does lead to drastic performance drop. This indicates that a residual network does not actually exhibit behaviors of an ultra-deep network, it rather behaves similarly to ensembles of many mutually independent shallow networks.

Width *vs*. Depth for ResNets. As discussed in Section 1 that the aggregation nature of shortcut connections links deep networks with such topology to pseudo-wide networks, we compare residual aggregation with actual wide

networks. We find that simply widening the network is of higher efficiency than stacking more residual blocks once the network has reached a certain depth. Reported by Zagoruyko *et al*. [40], a 40-layer residual network of 4× width outperformed a 1001-layer network on CIFAR with fewer parameters. A wider 101-layer residual network also achieved higher accuracy on ImageNet classification than a 200-layer network with the same model complexity [38]. One possible explanation is that residual aggregation entangles outputs from each layer and thus hinders the ability to search for new features [42]. We hence implement in-layer shortcut connections for SeqConv with concatenation instead of addition to avoid such limitation.

Aggregated transformations. The implementation of aggregated transformations is generally supported by a multi-path architecture. Each path applies a transformation with a small kernel and features produced by each path are then aggregated to formulate the final representation in a larger feature space. The representation capability is determined by the multi-path architecture and this could be, a set of homogeneous paths [38], a set of hierarchical paths [14], other more complex structures [39, 33], or even learnable structures as reflected by the cell design of recent studies on network architecture search [44, 16].

3. Methods

3.1. Sequentially Aggregated Transformations

Consider a regular wide convolutional layer gets divided into several groups of transformations, we employ a simple yet elegant hierarchical multi-path architecture to aggregate each group as briefly described in Section 1. A comparison between a standard convolutional layer and the proposed sequentially aggregated convolutional (SeqConv) layers is illustrated in Figure 1.

Basic layers. For a SeqConv layer with g groups of transformations, let x_0 denote the input of the layer, x_i denote the output of the i^{th} group. The layer is defined by:

$$x_i = F_i([x_0, x_1, \ldots, x_{i-1}]) \tag{1}$$

where F_i denotes the non-linear transformation function of the i^{th} group, while $[\ldots]$ refers to the concatenation operation. The final representation produced by a SeqConv layer is formulated by the aggregation of outputs from all g groups $[x_1, x_2, \ldots, x_g]$. The width of the representation is preserved by the concatenation-based aggregation while the depth is drastically increased. For each transformation function, the sequential aggregation enables the view over features extracted by previous groups. Each time a group of features x_i pass through the transformation function F_{i+1} of the following group, a group of deeper features x_{i+1} could be extracted. The final representation aggregates features of various depths, including very deep features from latter groups of the layer. The representation capability is thus greatly enhanced [5].

Relations to DenseNet. The sequential aggregation was first introduced to convolutional neural networks by DenseNet [14]. The hierarchical multi-path architecture defined by Eq.1 is shared by both SeqConv and a dense block in DenseNet, except that SeqConv does not include the input x_0 in its final representation similar to a regular convolutional layer while a dense block does. Aside from the apparent architectural similarity, SeqConv is still fundamentally different from DenseNet in two aspects:

- SeqConv is derived on a different basis than that of DenseNet. In [14], Huang *et al*. place heavy emphasis on feature reuse and improved information flow for deep networks using shortcut connections. SeqConv, instead, is based on the observation that it is not viable to build genuine deep networks using such network topology due to the vanishing gradient problem on the longest gradient path [36]. The aggregation nature and shorter gradient paths of shortcut connections already lead to behaviors resembling wide architectures in a seemingly deep network. We thus embrace this side effect that links shortcut connections to wide architectures to modify *actual* wide networks. Concretely, we utilize the hierarchical aggregation capability of shortcut connections as formulated by SeqConv to enhance the representational power of a single wide convolutional layer.

- SeqConv is derived from wide convolutional layers, and thus has a finer architectural granularity than DenseNet. SeqConv is a **layer-level** architecture and could be integrated into a large variety of backbone networks such as ResNet [11], ResNeXt [38], DLA [39], *etc*. by simply replacing the regular convolutional layers. Such flexibility has not been explored in [14] since each transformation unit in DenseNet is regarded as a separate layer. We argue that such interpretation not only impedes the flexibility to integrate sequential aggregation into other backbone networks, but itself might also be problematic. We further analyze this limitation in our following reinterpretation of DenseNet.

The layers in DenseNet are unusually narrow, the only rationale seems to be a vague statement about "collective knowledge" [14] and a clear analysis is absent. A layer, as described by [20], produces a representation of the input. This clearly is not the case of a "layer" in DenseNet. Features extracted by a DenseNet "layer" are not enough to solely constitute a representation and are instead always amalgamated with features extracted by other "layers" to

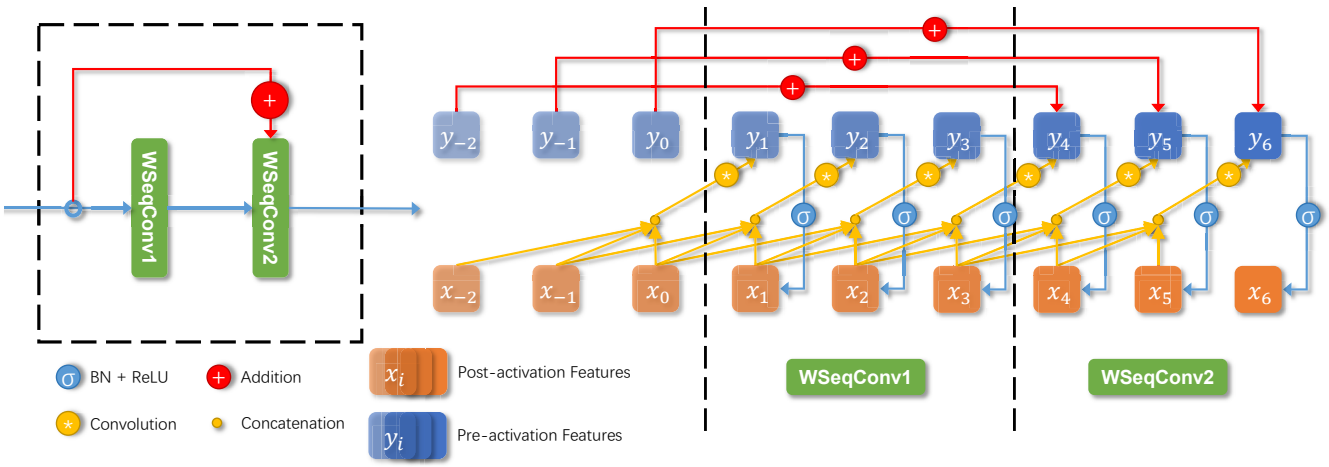

Figure 2. A residual block (left) and an exploded view of the block (right) revealing its internal structure in details.

jointly comprise a hierarchical representation. It thus might not be accurate to refer to such transformation units in DenseNet as "layers", since each one of them only contributes to one feature group of a very wide representation that comprises many such groups. We attribute the success of sequential aggregation, including DenseNet, to the expressiveness of hierarchical wide representations. Reuse of feature groups [14] is an implementation to produce such representations. Moreover, overemphasis on feature reuse also has its own drawbacks as we address in the following section of windowed SeqConv layers. It is also worth noting that DenseNet bears a close resemblance to SeqConv plugged into a VGG-like [26] backbone network.

Transformation functions. For a basic SeqConv layer, we follow the common settings in [10] and implement the transformation function with three consecutive operations: 3×3 convolution (Conv) followed by batch normalization (BN) [17] and a rectified linear unit (ReLU) [7]. The number of filters for each group is determined by the channel number of the convolution and is denoted by k.

To further improve the computational efficiency and model compactness, we also implement a bottleneck transformation function following [10, 35, 14]. For this bottleneck variant, we first employ a 1×1 Conv to reduce the channel number of the aggregated features down to k, then apply the transformation with a 3×3 Conv. The bottleneck transformation function comprises six consecutive operations: Conv(1×1)-BN-ReLU-Conv(3×3)-BN-ReLU, and the SeqConv layer with this transformation function is marked by the "B" postfix (SeqConvB).

Windowed layers. The dense aggregation defined by Eq.1 assigns more weights to earlier features of a representation produced by SeqConv. Consider a representation x'_0 of g' groups $[x_{1-g'}, x_{2-g'}, \ldots, x_0]$ produced by a previous SeqConv layer goes through the current layer. The earliest features $x_{1-g'}$ are utilized by both $F_{2-g'}, \ldots, F_0$ of the previous layer and all transformation functions of the current layer, while the latest features x_0 are only utilized by the current layer. $x_{1-g'}$ is thus assigned with more weights than x_0 since each time a group of features pass through a transformation function, certain weights are assigned to that group of features. The extra weights assigned to earlier features give rise to a vast number of required parameters growing at an asymptotic rate of $O(n^2)$, where n is the width of the SeqConv layer, whereas a regular convolutional layer merely has a linear parameter growth rate. Recent study [42] suggests that this quadratic growth suffers from significant parameter redundancy. It is observed that DenseNet, which shares the same aggregation mechanism with SeqConv, has many skip connections with average absolute weights close to zero [42, 13]. We also notice that features exploited by a particular group are mostly distributed over the outputs of recent preceding groups of that group [12, 42], since the information carried by the outputs of earlier groups has been abundantly exploited.

Thus, to reduce the parameter redundancy and lower the computational cost of SeqConv, we propose a windowed variant of SeqConv (WSeqConv) that only aggregates the outputs from most recent groups. The WSeqConv is defined as follows:

$$x_i = F_i([x_{i-g'}, x_{i-g'+1}, \ldots, x_{i-1}]) \tag{2}$$

the representation produced by WSeqConv is still formulated by the aggregation $[x_1, x_2, \ldots, x_g]$. Note that Eq.2 is equivalent to applying a sliding rectangular window function ϕ across the channel dimension on Eq.1:

$$\phi_i(x) = \begin{cases} 1 & i - g' \le x \le i - 1 \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

$$\omega_i = [\phi_i(1 - g'), \phi_i(2 - g'), \ldots, \phi_i(i - 1)] \tag{4}$$

$$
\begin{aligned}
x_i &= F_i([x_{i-g'}, x_{i-g'+1}, \ldots, x_{i-1}]) \\
&= F_i([x_{1-g'}, x_{2-g'}, \ldots, x_{i-1}] \circ \omega_i) \\
&= F_i([x'_0, x_1, \ldots, x_{i-1}] \circ \omega_i)
\end{aligned} \tag{5}
$$

$\circ$ denotes the operator for the Hadamard product. The window ϕ truncates the input for each group to a constant width, the parameter number and computational cost of WSeqConv are thus reduced to the same as of a regular convolutional layer.

3.2. Network Architecture

We apply SeqConv and WSeqConv to three widely-adopted backbone networks, pre-activation ResNet [11] with basic residual blocks, ResNet with bottleneck residual blocks and ResNeXt [38] and refer to them as, respectively, SeqResNet, SeqResNet-B and SeqResNeXt. We evaluate these models on image classification following [10, 38, 14, 42].

Residual blocks. We construct residual blocks with WSeqConv layers (or its bottleneck variant) as the building blocks of our networks. Following [10], we place two layers in each residual block. Residual connections are embedded in the aggregation of the second layer owing to pre-activation identity mapping as in [11]. This specific residual connectivity pattern allows earlier features that fall out of the aggregation view of the second layer to be implicitly shared with the layer without introducing any extra parameters, which further encourages feature reuse at the network level. A detailed breakdown of the structure of residual blocks with WSeqConv layers is illustrated in Figure 2.

Down-sampling blocks. Down-sampling is an essential part of the classification networks as it enables the network to extract features from different levels of abstraction. The common practice of down-sampling in a classification network reduces the spatial resolution of each feature map while the width of the representation is multiplied. This is however incompatible with sequential aggregation since it is not viable to aggregate feature maps of different spatial dimensions. We thus introduce down-sampling blocks to facilitate down-sampling for networks with SeqConv. A down-sampling block consists of an extension layer and a downsizing layer as illustrated in Figure 3. The representation is first extended to the target width by the extension layer. The spatial dimensions are then reduced by the downsizing layer. The grouping of downsizing prevents information leaks among features of different groups and hence preserves the hierarchy of the aggregated features, which is essential if the down-sampling block is followed by a WSeqConv layer that requires a hierarchical input.

Subgroups. A new dimension called "cardinality" was introduced in ResNeXt[38]. This dimension divides a 3×3 Conv into many small groups and allows for a wider representation compared to a regular convolutional layer with the same model complexity. To facilitate cardinality for SeqConv, we further divide the 3×3 Conv in the transformation function of each group into several subgroups by replacing it with a 3×3 grouped Conv. The number of subgroups for each group is denoted by c.

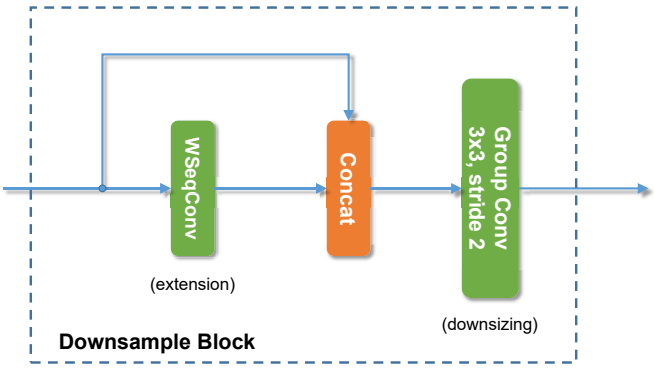

Figure 3. The topology of a down-sampling block, the extension layer and downsizing layer are, respectively, implemented by a WSeqConv layer and a grouped Conv layer of stride 2.

stage	output	SeqResNet	SeqResNet-B
conv1	32×32	3×3, 16	3×3, 16
conv2	32×32	3×3, 16×r *	1×1→3×3, 16×r *
		$\begin{bmatrix} 3\times3,\ 16\times r\ * \\ 3\times3,\ 16\times r\ * \end{bmatrix}\times N$	$\begin{bmatrix} 1\times1\to3\times3,\ 16\times r\ * \\ 1\times1\to3\times3,\ 16\times r\ * \end{bmatrix}\times N$
conv3	16×16	3×3, 16×r * Concatenate, 32×r 3×3, 32×r, stride 2, groups = $\frac{32\times r}{k}$	1×1→3×3, 16×r * Concatenate, 32×r 3×3, 32×r, stride 2, groups = $\frac{32\times r}{k}$
		$\begin{bmatrix} 3\times3,\ 32\times r\ * \\ 3\times3,\ 32\times r\ * \end{bmatrix}\times N$	$\begin{bmatrix} 1\times1\to3\times3,\ 32\times r\ * \\ 1\times1\to3\times3,\ 32\times r\ * \end{bmatrix}\times N$
conv4	8×8	3×3, 16×r * Concatenate, 48×r 3×3, 48×r, stride 2, groups = $\frac{48\times r}{k}$	1×1→3×3, 16×r * Concatenate, 48×r 3×3, 48×r, stride 2, groups = $\frac{48\times r}{k}$
		$\begin{bmatrix} 3\times3,\ 48\times r\ * \\ 3\times3,\ 48\times r\ * \end{bmatrix}\times N$	$\begin{bmatrix} 1\times1\to3\times3,\ 48\times r\ * \\ 1\times1\to3\times3,\ 48\times r\ * \end{bmatrix}\times N$
		1×1, 48×r	1×1, 48×r
	1×1	global average pool fc, softmax	global average pool fc, softmax

Table 1. Network template for CIFAR. We use a wide architecture following [11], the widening factor is denoted by r. N stands for the number of residual blocks for each stage. SeqConv and WSeqConv layers are marked by * and * respectively.

Implementation details. We build SeqResNet and SeqResNet-B for CIFAR based on [11]. The networks are divided into three stages with the feature map sizes of 32×32, 16×16, and 8×8 respectively. We place a regular Conv layer before the first SeqConv layer following [14], and an equal number of residual blocks for each stage. We insert a down-sampling block between each stage and

stage	output	SeqResNeXt-24	SeqResNet B42 / B22
conv1	112×112	3×3, 32, stride 2 3×3, 32	
conv2	56×56	1×1→3×3, 256, k=32, c=8 * 3×3, 256, stride 2, 64 groups	1×1→3×3, 128, k=32 * 3×3, 128, stride 2, 4 groups
		[1×1→3×3, 256, k=32, c=8 * 1×1→3×3, 256, k=32, c=8 *] ×1	[1×1→3×3, 128, k=32 * 1×1→3×3, 128, k=32 *] ×3 / ×1
conv3	28×28	1×1→3×3, 256, k=32, c=8 * concatenate, 512 3×3, 512, stride 2, 64 groups	1×1→3×3, 128, k=32 * concatenate, 256 3×3, 256, stride 2, 4 groups
		[1×1→3×3, 512, k=64, c=8 * 1×1→3×3, 512, k=64, c=8 *] ×1	[1×1→3×3, 256, k=64 * 1×1→3×3, 256, k=64 *] ×4 / ×1
conv4	14×14	1×1→3×3, 512, k=64, c=8 * concatenate, 1024 3×3, 1024, stride 2, 64 groups	1×1→3×3, 256, k=64 * concatenate, 512 3×3, 512, stride 2, 8 groups
		[1×1→3×3, 1024, k=64, c=4 * 1×1→3×3, 1024, k=64, c=4 *] ×3	[1×1→3×3, 512, k=64 * 1×1→3×3, 512, k=64 *] ×5 / ×2
conv5	7×7	1×1→3×3, 1024, k=64, c=4 * concatenate, 2048 3×3, 2048, stride 2, 64 groups	1×1→3×3, 512, k=64 * concatenate, 1024 3×3, 1024, stride 2, 8 groups
		[1×1→3×3, 2048, k=128, c=4 * 1×1→3×3, 2048, k=128, c=4 *] ×1	[1×1→3×3, 1024, k=128 * 1×1→3×3, 1024, k=128 *] ×3 / ×1
		1×1, 2048	1×1, 1024
	1×1	global average pool 1000-d fc, softmax	
# params.		26.2×10^6	25.6×10^6 / 11.8×10^6
GFLOPs		4.32	5.33 / 2.73

Table 2. Network architecture and model complexity of our ImageNet models. SeqResNeXt-24 and SeqResNet-B42 have a model complexity comparable to ResNet-50 while SeqResNet-B22 is about the half model size. SeqConv and WSeqConv layers are marked by * and * respectively.

a 1×1 convolution at the end of the third stage before global average pooling. A classifier consisting of a fully connected layer and a softmax activation is attached after average pooling. The exact specifications of our model template for CIFAR are listed in Table 1.

For the ImageNet evaluation, we adopt SeqResNet-B and SeqResNeXt with four stages on 224×224 inputs. We use different k for each stage of our ImageNet models due to the increasing model complexity and computational cost. Following [35, 37], We replace the expensive 7×7 convolution and the following max pooling with two 3×3 convolutions and a down-sampling block. We list the detailed configurations of our ImageNet models in Table 2.

4. Experiments

We conduct experiments on three public benchmark datasets: CIFAR-10, CIFAR-100 [18] and ImageNet [4]. We compare our models with their original backbones [10, 38] and similar state-of-the-art methods [14, 42].

4.1. Datasets

CIFAR. Both CIFAR-10 (C10) and CIFAR-100 [18] (C100) consist of 50,000 training samples and 10,000 test samples, which are divided into 10 and 100 classes respectively. All samples are color image of 32×32 pixels. We apply widely adopted data augmentation including mirroring and shifting as in [23, 25, 28, 21, 8] for these two datasets and normalize samples by the channel means and standard deviations. We refer to the augmented datasets as C10+ and C100+. 5,000 training samples are randomly selected for validation as we evaluate our models. We use all training samples for the final run following [14] and report the final test error at the end of training.

ImageNet. The ImageNet 2012 classification dataset [4] contains 1.28 million training images and 50,000 validation images drawn from 1,000 classes. We adopt the standard augmentation scheme following [8, 38, 14] and normalize the dataset by the channel means and standard deviations. We evaluate our models on the single 224×224 center crop following [11, 38].

4.2. Training

All models are optimized with stochastic gradient descent (SGD). We apply Nesterov momentum [32] of 0.9 and L2 weight regularization of 10^{-4} following [8]. We initialize the second WSeqConv layer of each residual block with zeros and all other Conv layers are initialized following [9]. The zero initialization disables all residual blocks and imitates a shallow network, which is easier to optimize at the initial stage of the training. Similar initialization procedures for ResNet are also proposed in [37, 41]. We apply the initialization introduced in [6] on the fully connected layer of the classifier. The training for all models starts with an initial learning rate of 0.1.

For the CIFAR datasets, we train our models for 300 epochs with batch size 64 and divide the learning rate by 10 at epoch 150 and 225. Due to the limited number of samples presented in these two datasets, we follow [12] and apply dropout [29] with a drop rate of 0.1 before the 1×1 convolution (prior to global average pooling) and every SeqConv layer (except the first one) to reduce overfitting.

The ImageNet models are trained for 100 epochs with batch size 256. We report the best validation error of the first 90 epochs of training and also the best error till all 100 epochs finish for a fair comparison with [10, 38, 14]. We reduce the learning rate by 10 times for every 30 epochs.

4.3. Results on CIFAR

The experimental results for CIFAR [18] are presented in Table 3. SeqResNet outperforms the corresponding ResNet baseline by a large margin. A SeqResNet-B with merely 0.8M parameters achieves higher accuracies than the 1001-layer ResNet counterpart with more than 10M parameters, which reduces the model complexity required to obtain an accuracy comparable to that of ResNet by a factor of **12**. SeqResNet also consistently outperforms the state-of-the-art

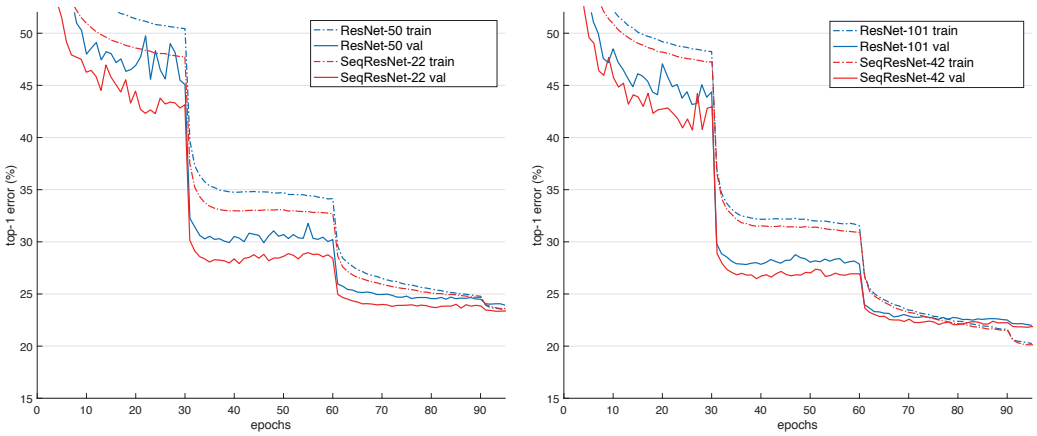

Figure 4. Training curves of SeqResNet-B22/ResNet-50 (left) and SeqResNet-B42/ResNet-101 (right) on ImageNet.

	settings	# params.	C10+	C100+
ResNet-110 [10]	-	1.7M	6.61	-
ResNet-110 (reported by [15])	-	1.7M	6.41	27.22
Wide ResNet-16 [40]	8× width	11.0M	4.81	22.07
Wide ResNet-28	10× width	36.5M	4.17	20.5
ResNet-164 (pre-activation) [11]	-	1.7M	5.46	24.33
ResNet-1001 (pre-activation)	-	10.2M	4.62	22.71
DenseNet [14]	$L = 40, k = 12$	1.1M	5.24	24.42
SparseNet [43]	$L = 40, k = 12$	0.8M	5.47	24.48
SeqResNet	$k = 8, r = 4, N = 1$	1.2M	**4.78**	**22.65**
DenseNet	$L = 100, k = 12$	7.2M	4.1	20.2
SparseNet	$L = 100, k = 16, 32, 64$	7.2M	4.21	19.89
SeqResNet	$k = 16, r = 10, N = 1$	7.6M	**3.97**	**19.72**
DenseNet-BC	$L = 100, k = 12$	0.8M	4.51	22.27
SparseNet-BC	$L = 100, k = 24$	1.5M	4.49	22.71
SeqResNet-B	$k = 16, r = 7, N = 1$	0.8M	**4.3**	**20.76**
DenseNet-BC	$L = 100, k = 16, 32, 64$	7.9M	4.02 *	19.55 *
SparseNet-BC	$L = 100, k = 16, 32, 64$	4.4M	4.34	19.9
SeqResNet-B	$k = 32, r = 12, N = 3$	6.0M	**3.72**	**18.51**

Table 3. Error rates (%) and model sizes on CIFAR. Results that surpass all competing methods are **bold**. * indicates results reported by [43].

DenseNet [14] and SparseNet [42] of similar model complexity. A SeqResNet with 1.2M parameters attains about 0.5% lower error rate on C10+ and 2% lower error rate on C100+ compared to its DenseNet and SparseNet counterparts. SeqResNet shows significantly higher parameter efficiency than wide architectures such as wide ResNet [40] and ResNeXt[38] as well. It only takes 6M parameters for SeqResNet-B to achieve higher accuracies than the 6× larger wide ResNet-28, or attain error rates comparable to the 5.73× larger ResNeXt-29, 8×64d (34.4M parameters, 3.65% error rate on C10+ and 17.77% on C100+).

4.4. Results on ImageNet

We evaluate SeqResNet-B and SeqResNeXt on the large-scale ILSVRC 2012 dataset to validate the scalability of our models. Table 4 reports the top-1 and top-5 validation errors of our models on ImageNet.

	# params.	FLOPs	top-1 err	top-5 err
ResNet-101 [10]	44.5M	7.34G	22.44	6.21
DenseNet-264 [14]	33.3M	5.52G	22.15	6.12
ResNet-152	60.2M	10.82G	22.16	6.16
SeqResNet-B42	25.6M	5.33G	**22.06**	**5.98**
SeqResNeXt-24	26.2M	4.32G	**21.92**	**5.82**
ResNet-50	25.6M	3.86G	24.01	7.02
DenseNet-169	14.1M	3.22G	23.80	6.85
SeqResNet-B22	11.8M	2.73G	**23.67**	**6.78**
ResNet-50 *	25.6M	3.86G	23.9	-
SeqResNet-B22 *	11.8M	2.73G	**23.35**	6.68
ResNeXt-50 [38] *	25.0M	4.00G	22.2	-
ResNet-101 *	44.5M	7.34G	22.0	-
SeqResNet-B42 *	25.6M	5.33G	**21.75**	5.89
SeqResNeXt-24 *	26.2M	4.32G	**21.50**	5.73

Table 4. Validation error rates on ImageNet. Models marked by * are trained for 100 epochs.

SeqResNet-B22 and SeqResNet-B42 not only surpass the performance of their ResNet counterpart of equal model size, but even go much further and outperform ResNets of significantly larger model complexity. Figure 4 (left) shows that SeqResNet-B22 with less than 12M parameters exhibits lower training error and validation error than the much larger ResNet-50 with more than 25M parameters. A similar trend between SeqResNet-B42 and ResNet-101 is also plotted in Figure 4 (right). The lower training error with much smaller model size indicates that SeqResNet has much stronger representational ability than ResNet, in fact, SeqResNet-B42 even outperforms the 2.35× larger ResNet-152. SeqResNet also shows superior performance compared against the state-of-the-art DenseNet [14] and ResNeXt [38]. Both SeqResNet models attain higher accuracy than their DenseNet counterpart with fewer parameters. The performance gap between SeqRetNet-B42 and ResNeXt-50 of similar model complexity, is marked by the 2× complexity ResNet-101 that ResNeXt-50 fails to outperform but surpassed by SeqResNet-B42.

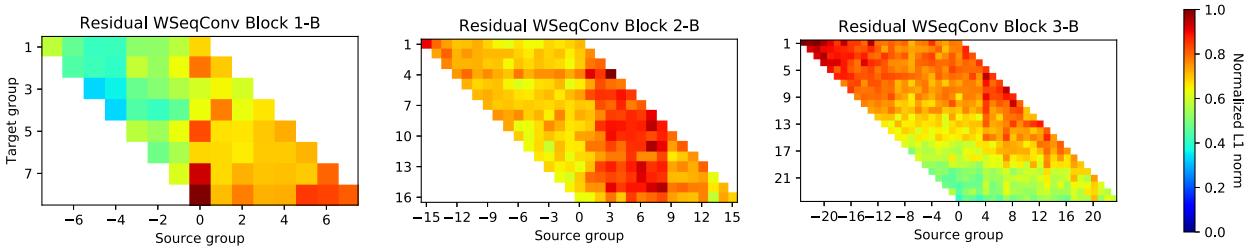

Figure 5. Weight visualization of three WSeqConv layers, we plot the heat map for the second WSeqConv layer of the residual block of each stage from a trained SeqResNet ($k = 8$, $r = 4$, $N = 1$).

Further performance gain could be observed on SeqResNeXt-24. It has a model complexity similar to SeqResNet-B42 but achieves higher accuracy with lower computational cost (FLOPs). SeqResNeXt-24 also significantly outperforms its ResNeXt-50 counterpart by a showing a top-1 error rate comparable to the $2\times$ complexity ResNeXt-101 (21.2%). To the best of our knowledge, SeqResNeXt-24 has the current best accuracy on ImageNet (with similar augmentation and training/testing procedures) for non-NAS [44] based models of similar model complexity (about 25M parameters).

4.5. Discussion

Hyperparameter Investigation. We empirically evaluate the effect of each hyperparameter as listed in Table 5. All comparing models have been adjusted to a similar complexity. $S1$ is the standard reference model as we present in Table 3. $S2$ disables windowed aggregation for all SeqConv layers, it thus has a smaller r (width) than $S1$ since a basic SeqConv layer has a higher complexity than WSeqConv. $S3$ adopts a larger k than $S1$. $S4$ further reduces r (width) and increases N (number of residual blocks) for $S3$. The higher error rate of $S2$ compared to $S1$ verifies the effectiveness of windowed aggregation. $S3$ has marginally more parameters than $S1$ while showing a higher error rate, which indicates that smaller k (more groups, deeper representation) might be beneficial to the representation capability. However, further performance gain on CIFAR with even smaller k is negligible. It is possible that the samples in the CIFAR dataset are too small (32×32) to utilize extremely deep features. Comparison between $S3$ and $S4$ shows that a wide network with fewer residual blocks performs better than a narrow network with more residual blocks.

Weight Visualization. We conduct the experiment proposed in [14] to visualize the weights of a trained SeqResNet. We pick 3 WSeqConv layers, each from a residual block of SeqResNet trained on C10+. We plot the weights based on the exploded view of SeqConv that the normalized weights corresponding to the connection between two groups are represented by a colored pixel. Results are plotted as heat maps in Figure 5. A red pixel indicates heavy use of an aggregated feature group while a blue pixel indicates low usage. Pixels of the white color indicate their corresponding feature group does not participate in the aggregation. We observe from the heat maps that there is hardly any blue pixel and a significant portion of the non-white pixels are red, indicating all parameters are reasonably exploited, whereas DenseNet [14] leaves large blue area on its heat maps. Our observation suggests that our model exhibits low parameter redundancy and fully exploits all aggregated features, which might explain the improved performance that our model attains with compact model size.

settings	r	k	N	windowed aggregation	# params.	err
$S1$	4	8	1	✓	1.2M	4.78
$S2$	3	8	{1, 2, 1}	✗	1.2M	4.90
$S3$	4	16	1	✓	1.3M	4.99
$S4$	3	16	{1, 2, 2}	✓	1.3M	5.15

Table 5. Error rates (%) of models with different hyperparameters on C10+.

5. Conclusion

We present a new form of aggregation-based convolutional layer (SeqConv) to enhance the representation capability of a single layer. SeqConv comprises various internal groups that are sequentially aggregated to extract features of various depths, and thus exhibits the benefits of both wide representation and deep representation. We also analyze the relations of SeqConv to DenseNet [14] which bears apparent similarity to our work, but is found to be ultimately different. A windowed aggregation mechanism is proposed as well to address the parameter redundancy of dense aggregation. SeqConv has the same model granularity as a regular convolutional layer and thus could be integrated into a wide variety of backbone networks. We adopt ResNet [11] and ResNeXt [38] as the backbone networks for our models. Experimental results on image classification indicate that our models with SeqConv significantly outperform their original backbones, and perform favorably against state-of-the-art methods [38, 14, 42].

References

[1] Sanjeev Arora, Simon S Du, Wei Hu, Zhiyuan Li, Ruslan Salakhutdinov, and Ruosong Wang. On exact computation with an infinitely wide neural net. *arXiv preprint arXiv:1904.11955*, 2019. 1

[2] Yoshua Bengio, Patrice Simard, and Paolo Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, 5(2):157–166, 1994. 1, 2

[3] Yunpeng Chen, Jianan Li, Huaxin Xiao, Xiaojie Jin, Shuicheng Yan, and Jiashi Feng. Dual path networks. In *Advances in Neural Information Processing Systems*, pages 4467–4475, 2017. 1, 2

[4] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Fei-Fei Li. Imagenet: a large-scale hierarchical image database. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009. 6

[5] Ronen Eldan and Ohad Shamir. The power of depth for feedforward neural networks. In *Annual Conference on Learning Theory*, volume 49, pages 907–940, 2016. 1, 3

[6] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *International Conference on Artificial Intelligence and Statistics*, volume 9, pages 249–256, 2010. 6

[7] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. In *International Conference on Artificial Intelligence and Statistics*, volume 15, pages 315–323, 2011. 4

[8] Sam Gross and Michael Wilber. Training and investigating residual nets. *Facebook AI Research*, 2016. 6

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *IEEE International Conference on Computer Vision*, pages 1026–1034, 2015. 6

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016. 1, 2, 4, 5, 6, 7

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *European Conference on Computer Vision*, pages 630–645. Springer, 2016. 2, 3, 5, 6, 7, 8

[12] Gao Huang, Shichen Liu, Laurens Van der Maaten, and Kilian Q Weinberger. Condensenet: An efficient densenet using learned group convolutions. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2752–2761, 2018. 2, 4, 6

[13] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. *arXiv preprint arXiv:1608.06993v5*, 2016. 4

[14] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 4700–4708, 2017. 1, 2, 3, 4, 5, 6, 7, 8

[15] Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Q Weinberger. Deep networks with stochastic depth. In *European Conference on Computer Vision*, pages 646–661. Springer, 2016. 2, 7

[16] Yanping Huang, Yonglong Cheng, Dehao Chen, HyoukJoong Lee, Jiquan Ngiam, Quoc V Le, and Zhifeng Chen. Gpipe: Efficient training of giant neural networks using pipeline parallelism. *arXiv preprint arXiv:1811.06965*, 2018. 3

[17] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*, volume 37, pages 448–456, 2015. 4

[18] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009. 6

[19] Gustav Larsson, Michael Maire, and Gregory Shakhnarovich. Fractalnet: Ultra-deep neural networks without residuals. In *International Conference on Learning Representations*, 2017. 1, 2

[20] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436, 2015. 3

[21] Chen-Yu Lee, Saining Xie, Patrick Gallagher, Zhengyou Zhang, and Zhuowen Tu. Deeply-supervised nets. In *International Conference on Artificial Intelligence and Statistics*, pages 562–570, 2015. 6

[22] Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. Visualizing the loss landscape of neural nets. In *Advances in Neural Information Processing Systems*, pages 6391–6401, 2018. 1

[23] Min Lin, Qiang Chen, and Shuicheng Yan. Network in network. In *International Conference on Learning Representations*, 2014. 6

[24] Zhou Lu, Hongming Pu, Feicheng Wang, Zhiqiang Hu, and Liwei Wang. The expressive power of neural networks: A view from the width. In *Advances in Neural Information Processing Systems*, pages 6231–6239, 2017. 1

[25] Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. Fitnets: Hints for thin deep nets. In *AISTATS*, 2015. 6

[26] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, 2015. 1, 4

[27] Pravendra Singh, Vinay Kumar Verma, Piyush Rai, and Vinay P Namboodiri. Hetconv: Heterogeneous kernel-based convolutions for deep cnns. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4835–4844, 2019. 2

[28] Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller. Striving for simplicity: The all convolutional net. *arXiv preprint arXiv:1412.6806*, 2014. 6

[29] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15:1929–1958, 2014. 6

[30] Rupesh K Srivastava, Klaus Greff, and Jürgen Schmidhuber. Highway networks. In *ICML workshops*, 2015. 1, 2

[31] Rupesh K Srivastava, Klaus Greff, and Jürgen Schmidhuber. Training very deep networks. In *Advances in Neural Information Processing Systems*, pages 2377–2385, 2015. 2

[32] Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. On the importance of initialization and momentum in deep learning. In *International Conference on Machine Learning*, pages 1139–1147, 2013. 6

[33] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *AAAI Conference on Artificial Intelligence*, 2017. 2, 3

[34] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–9, 2015. 1, 2

[35] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2818–2826, 2016. 4, 6

[36] Andreas Veit, Michael J Wilber, and Serge Belongie. Residual networks behave like ensembles of relatively shallow networks. In *Advances in Neural Information Processing Systems*, pages 550–558, 2016. 1, 2, 3

[37] Junyuan Xie, Tong He, Zhi Zhang, Hang Zhang, Zhongyue Zhang, and Mu Li. Bag of tricks for image classification with convolutional neural networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019. 6

[38] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1492–1500, 2017. 3, 5, 6, 7, 8

[39] Fisher Yu, Dequan Wang, Evan Shelhamer, and Trevor Darrell. Deep layer aggregation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2403–2412, 2018. 3

[40] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. In *British Machine Vision Conference*, pages 87.1–87.12, 2016. 1, 3, 7

[41] Hongyi Zhang, Yann N Dauphin, and Tengyu Ma. Fixup initialization: Residual learning without normalization. *arXiv preprint arXiv:1901.09321*, 2019. 6

[42] Ligeng Zhu, Ruizhi Deng, Michael Maire, Zhiwei Deng, Greg Mori, and Ping Tan. Sparsely aggregated convolutional networks. In *European Conference on Computer Vision*, pages 186–201, 2018. 2, 3, 4, 5, 6, 7, 8

[43] Ligeng Zhu, Ruizhi Deng, Michael Maire, Zhiwei Deng, Greg Mori, and Ping Tan. Sparsely aggregated convolutional networks. *arXiv preprint arXiv:1801.05895v3*, 2018. 7

[44] Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V Le. Learning transferable architectures for scalable image recognition. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8697–8710, 2018. 3, 8