FRACTALNET: ULTRA-DEEP NEURAL NETWORKS WITHOUT RESIDUALS

Gustav Larsson
University of Chicago
larsson@cs.uchicago.edu

Michael Maire
TTI Chicago
mmaire@ttic.edu

Gregory Shakhnarovich
TTI Chicago
greg@ttic.edu

ABSTRACT

We introduce a design strategy for neural network macro-architecture based on self-similarity. Repeated application of a simple expansion rule generates deep networks whose structural layouts are precisely truncated fractals. These networks contain interacting subpaths of different lengths, but do not include any pass-through or residual connections; every internal signal is transformed by a filter and nonlinearity before being seen by subsequent layers. In experiments, fractal networks match the excellent performance of standard residual networks on both CIFAR and ImageNet classification tasks, thereby demonstrating that residual representations may not be fundamental to the success of extremely deep convolutional neural networks. Rather, the key may be the ability to transition, during training, from effectively shallow to deep. We note similarities with student-teacher behavior and develop drop-path, a natural extension of dropout, to regularize co-adaptation of subpaths in fractal architectures. Such regularization allows extraction of high-performance fixed-depth subnetworks. Additionally, fractal networks exhibit an anytime property: shallow subnetworks provide a quick answer, while deeper subnetworks, with higher latency, provide a more accurate answer.

我们为神经网络的宏观架构引进了一种基于自相似的设计策略。单一扩展规则的重复应用生成一个极深的网络，其结构布局正是一个截断的分形。这样的网络包含长度不同的相互作用子路径，但不包含任何直通式连接（pass-through connections)：每个内部信号在被下一层看见之前，都被一个过滤器和非线性部件变换过了。这个性质与当前显式规划极深网络的结构使得训练成为一个残差学习问题的方法形成鲜明对比。我们的实验表明，残差表示不是极深卷积神经网络成功的基本要素。分形设计在 CIFAR-100 数据集上达到 22.85% 的错误率，与残差网络的最佳水平持平。

除了高性能之外，分形网络还展现了有趣的性质。它们可被认为是各种深度的子网络的隐式联合且计算效率高。我们探讨了对于训练的影响，简单提及了与“师徒”行为的联系，最重要的，展示了提取高性能固定深度子网络的能力。为了实现后一个任务，我们发展路径舍弃方法（drop-path; 对 dropout 的自然扩展）正则化分形架构里子路径的协同适应。通过这样的正则化，分形网络展示了一种无例外的性质：浅层子网络给出快捷的答案，而深度子网络（有较多延迟）给出更精准的答案。

1 INTRODUCTION

Residual networks (He et al., 2016a), or ResNets, lead a recent and dramatic increase in both depth and accuracy of convolutional neural networks, facilitated by constraining the network to learn residuals. ResNet variants (He et al., 2016a;b; Huang et al., 2016b) and related architectures (Srivastava et al., 2015) employ the common technique of initializing and anchoring, via a pass-through channel, a network to the identity function. Training now differs in two respects. First, the objective changes to learning residual outputs, rather than unreferenced absolute mappings. Second, these networks exhibit a type of deep supervision (Lee et al., 2014), as near-identity layers effectively reduce distance to the loss. He et al. (2016a) speculate that the former, the residual formulation itself, is crucial.

最近的 ResNet 在深度和精度上比卷积神经网络（CNN）做出了极大进步，方法是让网络对残差进行学习。ResNet 的变型以及类似的架构通过直接通道（pass-through channel；相当于恒等函数的网络）使用了共同的初始化和锚定技术。这样，训练有两个方面不同。第一，目标变成对残差输出的学习而不是对未被提及的绝对映射学习。第二，这些网络展示了一种深度监督，因为接近恒等的层有效减少了损失距离。何恺明等人 [8] 推测前者（即残差的构造）是关键的。

We show otherwise, by constructing a competitive extremely deep architecture that does not rely on residuals. Our design principle is pure enough to communicate in a single word, fractal, and a simple diagram (Figure 1). Yet, fractal networks implicitly recapitulate many properties hard-wired into previous successful architectures. Deep supervision not only arises automatically, but also drives a type of student-teacher learning (Ba & Caruana, 2014; Urban et al., 2017) internal to the network. Modular building blocks of other designs (Szegedy et al., 2015; Liao & Carneiro, 2015) resemble special cases of a fractal network's nested substructure.

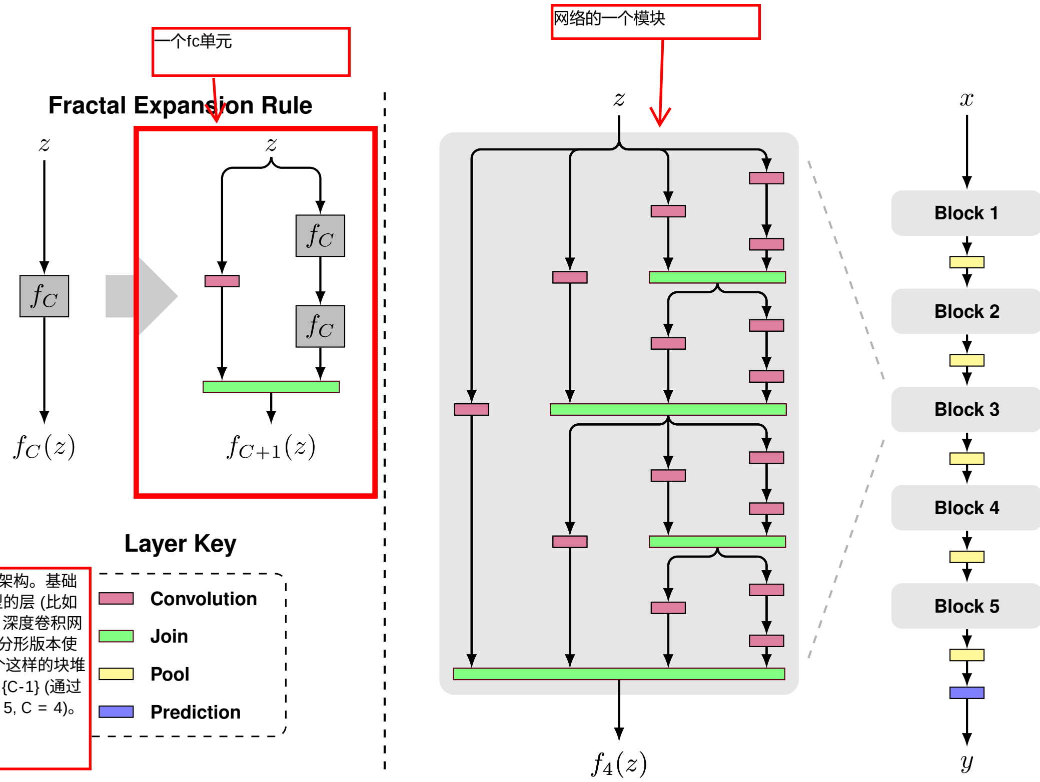
答案并非如此。我们通过构造不依赖于残差的极深架构揭示了这一点。我们的设计原则非常单纯，用“分形”一个词以及一幅图（图 1）就足以描述。可是，分形网络隐含地重现了过去的成功架构里硬性加入的许多性质。深度监督不仅自然出现，并且驱动了一种网络内部的“师徒学习”（student-teacher learning）。其他设计的模块化构成单元几乎都是分形网络嵌套子结构的特殊情形。

For fractal networks, simplicity of training mirrors simplicity of design. A single loss, attached to the final layer, suffices to drive internal behavior mimicking deep supervision. Parameters are randomly initialized. As they contain subnetworks of many depths, fractal networks are robust to choice of overall depth; make them deep enough and training will carve out a useful assembly of subnetworks.

对于分形网络，训练的简单性与设计的简单性相对应。单个连接到最后一层的损失函数足以驱动内部行为去模仿深度监督。参数是随机初始化的。由于包含不同深度的子网络，分形网络对总体深度的选取不敏感；让深度足够，然后训练会刻划出有用的子网络集合。

The entirety of emergent behavior resulting from a fractal design may erode the need for recent engineering tricks intended to achieve similar effects. These tricks include residual functional forms with identity initialization, manual deep supervision, hand-crafted architectural modules, and student-teacher training regimes. Section 2 reviews this large body of related techniques. Hybrid designs could certainly integrate any of them with a fractal architecture; we leave open the question of the degree to which such hybrids are synergistic.

分形设计导致的一系列涌现行为可能会让近期为了达到类似效果发展出的工程技巧变得不那么必要。这些技巧包括，恒等初始化的残差函数形式，手动深度监督，手工雕琢的架构模块，以及师徒训练体系。第 2 节回顾了这些相互关联的技术。混合设计当然可以把它们中任何一个与分形架构集成；关于这种混合体在多大程度上是多余或是互相促进这个问题，我们持开放态度。



分形架构。左：简单的扩展规则生成分形架构。基础情形f1(z) 包含输入输出之间单个选定类型的层 (比如卷积层)。连接层计算了逐元素平均。右：深度卷积网络周期性地通过汇集来降低空间分辨率。分形版本使用fC 作为汇集层之间的建构单元。把 B 个这样的块堆起来就得到一个网络，其总深度是 B * 2^{C-1} (通过卷积层衡量)。这个例子的深度是 40 (B = 5, C = 4)。

Figure 1: **Fractal architecture.** *Left:* A simple expansion rule generates a fractal architecture with C intertwined columns. The base case, $f_1(z)$, has a single layer of the chosen type (*e.g.* convolutional) between input and output. Join layers compute element-wise mean. *Right:* Deep convolutional networks periodically reduce spatial resolution via pooling. A fractal version uses f_C as a building block between pooling layers. Stacking B such blocks yields a network whose total depth, measured in terms of convolution layers, is $B \cdot 2^{C-1}$. This example has depth 40 ($B = 5$, $C = 4$).

Our main contribution is twofold:

1. 引入了 FractalNet，第一个在极深卷积神经网络领域对于 ResNet 的替代品。FractalNet 在科学上令人吃惊，它表明了残差学习对于极深网络不是必需的。

2. 通过分析和实验，我们阐释了 FractalNet 与一系列被加入到之前深度网络设计的那些现象之间的联系。

- We introduce FractalNet, the first simple alternative to ResNet. FractalNet shows that explicit residual learning is not a requirement for building ultra-deep neural networks.
- Through analysis and experiments, we elucidate connections between FractalNet and an array of phenomena engineered into previous deep network designs.

还有一个贡献，我们为极深网络开发了一种称为路径舍弃的新的正则化规则。已有的关于深度残差网络的工作其实缺乏有效正则化技术的展示，而仅仅依赖于数据增强。无需数据增强，通过 dropout 和路径舍弃训练的分形网络远远超过了报道的残差网络的性能。

As an additional contribution, we develop drop-path, a novel regularization protocol for ultra-deep fractal networks. Without data augmentation, fractal networks, trained with drop-path and dropout (Hinton et al., 2012), exceed the performance of residual networks regularized via stochastic depth (Huang et al., 2016b). Though, like stochastic depth, it randomly removes macro-scale components, drop-path further exploits our fractal structure in choosing which components to disable.

路径舍弃不仅是一种直观的正则化策略，而且提供了保证训练出的分形网络拥有无例外性质的方法。具体来说，训练期间对路径舍弃的特定安排阻止了不同深度的子网络的协同适应。结果就是，浅和深的网络必需各自能输出正确结果。测试期间，对浅子网络的请求在整个网络完成之前给出快速而精度普通的答案。

Drop-path constitutes not only a regularization strategy, but also provides means of optionally imparting fractal networks with anytime behavior. A particular schedule of dropped paths during learning prevents subnetworks of different depths from co-adapting. As a consequence, both shallow and deep subnetworks must individually produce correct output. Querying a shallow subnetwork thus yields a quick and moderately accurate result in advance of completion of the full network.

第 3 节给出了分形网络和路径舍弃的细节。第 4 节提供了基于 CIFAR-10, CIFAR-100 以及 SVHN 数据集与残差网络的比较。我们还评估了正则化和数据增强策略，探究了训练期间的子网络师徒行为，评价了通过路径舍弃得到的 anytime 网络。第 5 节提供了综合。通过容纳许多已知但看起来不同的设计原理，自相似结构也许会变成神经网络架构的基本组件。

Section 3 elaborates the technical details of fractal networks and drop-path. Section 4 provides experimental comparisons to residual networks across the CIFAR-10, CIFAR-100 (Krizhevsky, 2009), SVHN (Netzer et al., 2011), and ImageNet (Deng et al., 2009) datasets. We also evaluate regularization and data augmentation strategies, investigate subnetwork student-teacher behavior during training, and benchmark anytime networks obtained using drop-path. Section 5 provides synthesis. By virtue of encapsulating many known, yet seemingly distinct, design principles, self-similar structure may materialize as a fundamental component of neural architectures.

2 RELATED WORK

Deepening feed-forward neural networks has generally returned dividends in performance. A striking example within the computer vision community is the improvement on the ImageNet (Deng et al., 2009) classification task when transitioning from AlexNet (Krizhevsky et al., 2012) to VGG (Simonyan & Zisserman, 2015) to GoogLeNet (Szegedy et al., 2015) to ResNet (He et al., 2016a). Unfortunately, greater depth also makes training more challenging, at least when employing a first-order optimization method with randomly initialized layers. As the network grows deeper and more non-linear, the linear approximation of a gradient step becomes increasingly inappropriate. Desire to overcome these difficulties drives research on both optimization techniques and network architectures.

On the optimization side, much recent work yields improvements. To prevent vanishing gradients, ReLU activation functions now widely replace sigmoid and tanh units (Nair & Hinton, 2010). This subject remains an area of active inquiry, with various tweaks on ReLUs, *e.g.* PReLUs (He et al., 2015), and ELUs (Clevert et al., 2016). Even with ReLUs, employing batch normalization (Ioffe & Szegedy, 2015) speeds training by reducing internal covariate shift. Good initialization can also ameliorate this problem (Glorot & Bengio, 2010; Mishkin & Matas, 2016). Path-SGD (Neyshabur et al., 2015) offers an alternative normalization scheme. Progress in optimization is somewhat orthogonal to our architectural focus, with the expectation that advances in either are ripe for combination.

Notable ideas in architecture reach back to skip connections, the earliest example of a nontrivial routing pattern within a neural network. Recent work further elaborates upon them (Maire et al., 2014; Hariharan et al., 2015). Highway networks (Srivastava et al., 2015) and ResNet (He et al., 2016a;b) offer additional twists in the form of parameterized pass-through and gating. In work subsequent to our own, Huang et al. (2016a) investigate a ResNet variant with explicit skip connections. These methods share distinction as the only other designs demonstrated to scale to hundreds of layers and beyond. ResNet's building block uses the identity map as an anchor point and explicitly parameterizes an additive correction term (the residual). Identity initialization also appears in the context of recurrent networks (Le et al., 2015). A tendency of ResNet and highway networks to fall-back to the identity map may make their effective depth much smaller than their nominal depth.

Some prior results hint at what we experimentally demonstrate in Section 4. Namely, reduction of effective depth is key to training extremely deep networks; residuals are incidental. Huang et al. (2016b) provide one clue in their work on stochastic depth: randomly dropping layers from ResNet during training, thereby shrinking network depth by a constant factor, provides additional performance benefit. We build upon this intuition through drop-path, which shrinks depth much more drastically.

The success of deep supervision (Lee et al., 2014) provides another clue that effective depth is crucial. Here, an auxiliary loss, forked off mid-level layers, introduces a shorter path during backpropagation. The layer at the fork receives two gradients, originating from the main loss and the auxiliary loss, that are added together. Deep supervision is now common, being adopted, for example, by GoogLeNet (Szegedy et al., 2015). However, irrelevance of the auxiliary loss at test time introduces the drawback of having a discrepancy between the actual objective and that used for training.

Exploration of the student-teacher paradigm (Ba & Caruana, 2014) illuminates the potential for interplay between networks of different depth. In the model compression scenario, a deeper network (previously trained) guides and improves the learning of a shallower and faster student network (Ba & Caruana, 2014; Urban et al., 2017). This is accomplished by feeding unlabeled data through the teacher and having the student mimic the teacher's soft output predictions. FitNets (Romero et al., 2015) explicitly couple students and teachers, forcing mimic behavior across several intermediate points in the network. Our fractal networks capture yet another alternative, in the form of implicit coupling, with the potential for bidirectional information flow between shallow and deep subnetworks.

Widening networks, by using larger modules in place of individual layers, has also produced performance gains. For example, an Inception module (Szegedy et al., 2015) concatenates results of convolutional layers of different receptive field size. Stacking these modules forms the GoogLeNet architecture. Liao & Carneiro (2015) employ a variant with maxout in place of concatenation. Figure 1 makes apparent our connection with such work. As a fractal network deepens, it also widens. Moreover, note that stacking two 2D convolutional layers with the same spatial receptive field (*e.g.* 3×3) achieves a larger (5×5) receptive field. A horizontal cross-section of a fractal network is reminiscent of an Inception module, except with additional joins due to recursive structure.

3 FRACTAL NETWORKS

我们从对图 1 中描绘的想法的正式陈述开始。我们将以卷积神经网络作为例子和实验平台。但是，需要强调我们的框架更有一般性。原则上，为了生成其他分形架构，图 1 中的卷积层可以被替换为不同的层类型，甚至是定制化的模块或子网络。

We begin with a more formal presentation of the ideas sketched in Figure 1. Convolutional neural networks serve as our running example and, in the subsequent section, our experimental platform. However, it is worth emphasizing that our framework is more general. In principle, convolutional layers in Figure 1 could be replaced by a different layer type, or even a custom-designed module or subnetwork, in order to generate other fractal architectures.

令 C 表示截断分形 fC(.) 的指标。我们的网络结构、连接以及层类型，通过f C (.) 定义。包含单个卷积层的网络是基础情形：

Let C denote the index of the truncated fractal $f_C(\cdot)$. Our network's structure, connections and layer types, is defined by $f_C(\cdot)$. A network consisting of a single convolutional layer is the base case:

$$f_1(z) = \mathrm{conv}(z) \tag{1}$$

We define successive fractals recursively:

$$f_{C+1}(z) = [(f_C \circ f_C)(z)] \oplus [\mathrm{conv}(z)] \tag{2}$$

这里 o 表示复合，而 ⊕ 表示连接操作。当以图 1 的风格来画时，C 对应于列数，或者说网络 fC(.) 的宽度。深度定义为从输入到输出的最长路径上的 conv 层的个数，正比于 2^{C-1}。用于分类的卷积网络通常分散布置汇集层。为了达到相同目的，我们使用fC(.) 作为构建单元，将之与接下来的汇集层堆叠 B 次，得到总深度 B * 2^{C-1}。

where $\circ$ denotes composition and $\oplus$ a join operation. When drawn in the style of Figure 1, C corresponds to the number of columns, or width, of network $f_C(\cdot)$. Depth, defined to be the number of conv layers on the longest path between input and output, scales as 2^{C-1}. Convolutional networks for classification typically intersperse pooling layers. We achieve the same by using $f_C(\cdot)$ as a building block and stacking it with subsequent pooling layers B times, yielding total depth $B \cdot 2^{C-1}$.

连接操作 ⊕ 把两个特征块合为一个。这里，一个特征块是一个 conv 层的结果：在一个空间区域为固定的一些通道维持活化的张量。通道数对应于前面的 conv 层的过滤器的个数。当分形被扩展，我们把相邻的连接合并成单个连接层；如图 1 右侧所示，这个连接层跨越多列。连接层把所有其输入特征块合并成单个输出块。

The join operation $\oplus$ merges two feature blobs into one. Here, a blob is the result of a conv layer: a tensor holding activations for a fixed number of channels over a spatial domain. The channel count corresponds to the size of the filter set in the preceding conv layer. As the fractal is expanded, we collapse neighboring joins into a single join layer which spans multiple columns, as shown on the right side of Figure 1. The join layer merges all of its input feature blobs into a single output blob.

连接层行为的几种选择看起来都是合理的，包括拼接和加法。我们把每个连接实例化，计算其输入的逐元素平均。这对于卷积网络是恰当的，在那里通道数对一个分形块里的所有 conv 层是相同的。平均操作可能看起来类似 ResNet 的加法操作，但有几个关键不同：

Several choices seem reasonable for the action of a join layer, including concatenation and addition. We instantiate each join to compute the element-wise mean of its inputs. This is appropriate for convolutional networks in which channel count is set the same for all conv layers within a fractal block. Averaging might appear similar to ResNet's addition operation, but there are critical differences:

- ResNet makes clear distinction between pass-through and residual signals. In FractalNet, no signal is privileged. Every input to a join layer is the output of an immediately preceding conv layer. The network structure alone cannot identify any as being primary.
- Drop-path regularization, as described next in Section 3.1, forces each input to a join to be individually reliable. This reduces the reward for even implicitly learning to allocate part of one signal to act as a residual for another.
- Experiments show that we can extract high-performance subnetworks consisting of a single column (Section 4.2). Such a subnetwork is effectively devoid of joins, as only a single path is active throughout. They produce no signal to which a residual could be added.

ResNet 明确区分了直接通过与残差信号。在 FractalNet 里，没有什么信号是优越的。每一个对联合层的输入是上一个 conv 层的输出。单凭网络结构本身不能识别什么是主要的。

路径舍弃正则化，如第 3.1 节描述，强制让每个连接层的输入独自可靠。这降低了回报，即便是隐式学习把一个信号的一部分作为另一个的残差这种情形。

实验表明我们可以提取仅包含一列的高性能子网络 (4.2 节)。这样的子网络实际上不包含连接。它们不提供可以与残差相加的信号。

这些性质保证了连接层不是残差学习的一个替代方法。

Together, these properties ensure that join layers are not an alternative method of residual learning.

3.1 REGULARIZATION VIA DROP-PATH

Dropout 和 drop-connect 通过修改网络层序列之间的相互作用来减弱共同适应 (co-adaptation)。由于分形网络包含额外的大尺度结构，我们提出使用一种类似的粗粒度正则化策略来辅助这些技术

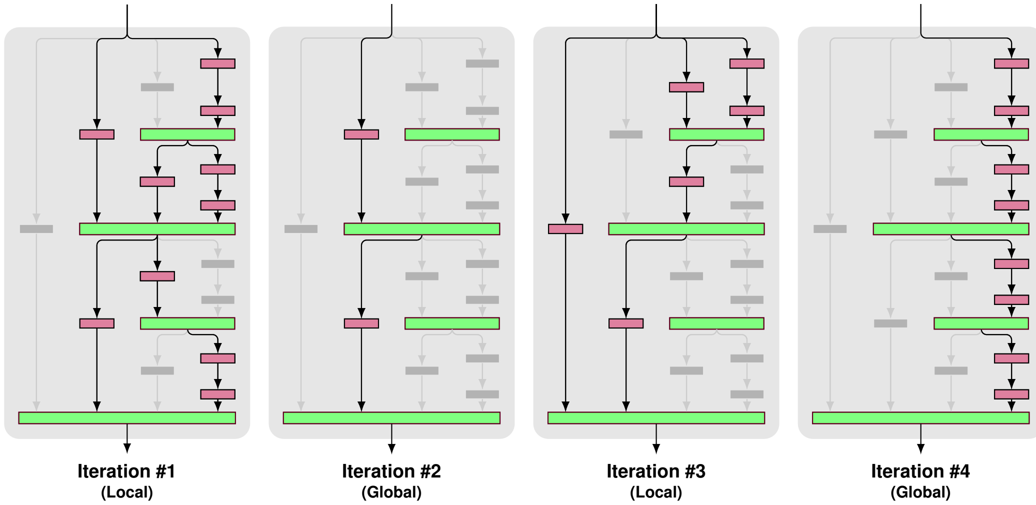
Dropout (Hinton et al., 2012) and drop-connect (Wan et al., 2013) modify interactions between sequential network layers in order to discourage co-adaptation. Since fractal networks contain additional macro-scale structure, we propose to complement these techniques with an analogous coarse-scale regularization scheme.

图 2 解释了路径舍弃。如同 dropout 禁止了活化的共同适应，路径舍弃通过随机丢弃连接层的操作数来禁止平行路径的共同适应。这压制了网络使用一个路径作为锚标，用另一个作为修正 (这可能导致过拟合) 的行为。我们考虑两个采样策略：

Figure 2 illustrates drop-path. Just as dropout prevents co-adaptation of activations, drop-path prevents co-adaptation of parallel paths by randomly dropping operands of the join layers. This discourages the network from using one input path as an anchor and another as a corrective term (a configuration that, if not prevented, is prone to overfitting). We consider two sampling strategies:

1. Local：对join层的输入dropout，但是至少保证要有一个输入；
2. Global: 对于整个网络来说，只选择一条路径，且限制为某个单独列，所以这条路径是独立的强预测路径；

- **Local**: a join drops each input with fixed probability, but we make sure at least one survives.
- **Global**: a single path is selected for the entire network. We restrict this path to be a single column, thereby promoting individual columns as independently strong predictors.



分形网络。某些层之间的连接被停用后仍能工作，前提是从输入到输出的某些路径还存在。路径舍弃保证至少一条这样的路径存在，与此同时许多其它路径被停用。训练期间，对每个mini-batch 展示不同的活跃子网络阻止了平行路径的共同适应。全局采样策略返回单列作为子网络。与局部采样交替使用鼓励了每列发展为表现良好的单独运作的子网络。

Figure 2: **Drop-path.** A fractal network block functions with some connections between layers disabled, provided some path from input to output is still available. Drop-path guarantees at least one such path, while sampling a subnetwork with many other paths disabled. During training, presenting a different active subnetwork to each mini-batch prevents co-adaptation of parallel paths. A global sampling strategy returns a single column as a subnetwork. Alternating it with local sampling encourages the development of individual columns as performant stand-alone subnetworks.

如同 dropout，信号可能需要恰当的缩放。对于逐元素平均，这是平凡的；连接仅仅计算了活跃输入的平均。

As with dropout, signals may need appropriate rescaling. With element-wise means, this is trivial; each join computes the mean of only its active inputs.

实验中，我们使用了 dropout 以及对路径舍弃采用了 50% 局部以及 50% 全局的混合采样。我们在每个 mini-batch 采样一个新的子网络。内存足够的情况下，对每个 mini-batch，我们可以同时对一个局部样本和所有全局样本求值。

In experiments, we train with dropout and a mixture model of 50% local and 50% global sampling for drop-path. We sample a new subnetwork each mini-batch. With sufficient memory, we can simultaneously evaluate one local sample and all global samples for each mini-batch by keeping separate networks and tying them together via weight sharing.

While fractal connectivity permits the use of paths of any length, global drop-path forces the use of many paths whose lengths differ by orders of magnitude (powers of 2). The subnetworks sampled by drop-path thus exhibit large structural diversity. This property stands in contrast to stochastic depth regularization of ResNet, which, by virtue of using a fixed drop probability for each layer in a chain, samples subnetworks with a concentrated depth distribution (Huang et al., 2016b).

全局路径舍弃不仅仅是作为一个正则化器，也是一个诊断工具。监控单列的表现体提供了关于网络和训练机制的洞察，这将在 4.3 节进一步提及。单个强大的列也让使用者在速度和精度方面取舍。

Global drop-path serves not only as a regularizer, but also as a diagnostic tool. Monitoring performance of individual columns provides insight into both the network and training mechanisms, as Section 4.3 discusses in more detail. Individually strong columns of various depths also give users choices in the trade-off between speed (shallow) and accuracy (deep).

3.2 Data Augmentation

数据增强可以显著降低对正则化的需求。ResNet 展示了这一点，对 CIFAR-100 数据集从 44.76%的错误率降低到 27.22%。虽然数据增强对分形网络有好处，我们提出一点，那就是路径舍弃提供了高度有效的正则化，让它们无需数据增强也能达到有竞争力的结果。

Data augmentation can reduce the need for regularization. ResNet demonstrates this, achieving 27.22% error rate on CIFAR-100 with augmentation compared to 44.76% without (Huang et al., 2016b). While augmentation benefits fractal networks, we show that drop-path provides highly effective regularization, allowing them to achieve competitive results even without data augmentation.

3.3 Implementation Details

我们使用 Caffe 实现 FractalNet。纯粹为了实现的方便，我们把图 1 中汇集和连接层的顺序翻转了。每列在跨越所有列的连接层之前单独汇集，而不是在那之后。

We implement FractalNet using Caffe (Jia et al., 2014). Purely for convenience, we flip the order of pool and join layers at the end of a block in Figure 1. We pool individual columns immediately before the joins spanning all columns, rather than pooling once immediately after them.

我们使用含冲量的随机梯度下降来训练分形网络。我们对每个 conv 层引进批量归一化（卷积，批量归一化，ReLU）。

We train fractal networks using stochastic gradient descent with momentum. As now standard, we employ batch normalization together with each conv layer (convolution, batch norm, then ReLU).

表 1 报告了 FractalNet 的表现，对应 C = 3 列，应用到 CIFAR 和 SVHN 分类任务上。也列了所有竞争方法的分数。整个表上 FractalNet 超过了原始 ResNet 的表现。使用数据增强后，FractalNet 在 CIFAR-100 上的表现几乎跟 ResNet 的最佳变型一模一样。不使用数据增强和正则化，我们模型在 CIFAR-100 和 CIFAR-10 上的表现优于 ResNet 和 随机深度的 ResNet。这提示与 ResNet 相比FractalNet 可能不那么容易过拟合。所有方法在 SVHN 上表现都不错，因此那个数据集没什么区分度。

Method	C100	C100+	C100++	C10	C10+	C10++	SVHN
Network in Network (Lin et al., 2013)	35.68	-	-	10.41	8.81	-	2.35
Generalized Pooling (Lee et al., 2016)	32.37	-	-	7.62	6.05	-	1.69
Recurrent CNN (Liang & Hu, 2015)	31.75	-	-	8.69	7.09	-	1.77
Multi-scale (Liao & Carneiro, 2015)	27.56	-	-	6.87	-	-	1.76
FitNet Romero et al. (2015)	-	35.04	-	-	8.39	-	2.42
Deeply Supervised (Lee et al., 2014)	-	34.57	-	9.69	7.97	-	1.92
All-CNN (Springenberg et al., 2014)	-	33.71	-	9.08	7.25	4.41	-
Highway Net (Srivastava et al., 2015)	-	32.39	-	-	7.72	-	-
ELU (Clevert et al., 2016)	-	24.28	-	-	6.55	-	-
Scalable BO (Snoek et al., 2015)	-	-	27.04	-	-	6.37	1.77
Fractional Max-Pool (Graham, 2014)	-	-	26.32	-	-	3.47	-
FitResNet (Mishkin & Matas, 2016)	-	27.66	-	-	5.84	-	-
ResNet (He et al., 2016a)	-	-	-	-	6.61	-	-
ResNet by (Huang et al., 2016b)	44.76	27.22	-	13.63	6.41	-	2.01
Stochastic Depth (Huang et al., 2016b)	37.80	24.58	-	11.66	5.23	-	1.75
Identity Mapping (He et al., 2016b)	-	22.68	-	-	4.69	-	-
ResNet in ResNet (Targ et al., 2016)	-	22.90	-	-	5.01	-	-
Wide (Zagoruyko & Komodakis, 2016)	-	20.50	-	-	4.17	-	-
DenseNet-BC (Huang et al., 2016a)[1]	19.64	17.60	-	5.19	3.62	-	1.74
FractalNet (20 layers, 38.6M params)	35.34	23.30	22.85	10.18	5.22	5.11	2.01
+ drop-path + dropout	28.20	23.73	23.36	7.33	4.60	4.59	1.87
↳ deepest column alone	29.05	24.32	23.60	7.27	4.68	4.63	1.89
FractalNet (40 layers, 22.9M params)[2]	-	22.49	21.49	-	5.24	5.21	-

Table 1: **CIFAR-100/CIFAR-10/SVHN.** We compare test error (%) with other leading methods, trained with either no data augmentation, translation/mirroring (+), or more substantial augmentation (++). Our main point of comparison is ResNet. We closely match its benchmark results using data augmentation, and outperform it by large margins without data augmentation. Training with drop-path, we can extract from FractalNet single-column (plain) networks that are highly competitive.

4 Experiments

我们采用 CIFAR-100, CIFAR-10 以及 SVHN 作为比较用的数据集。我们评估在每个数据集的标准分类任务上的表现。这些数据集都由 32 x 32 图片组成，所以我们固定分形网络的块数为 5 (B=5)，采用 2 x 2 的不重叠的 max-pooling。这就把输入的 32 x 32 分辨率变成 1 x 1。网络终端连接了 softmax 的预测层。

除非专门提及，我们把 5 个块中过滤器的个数设为（64,128, 256, 512, 512）。这个递增数列类似于 VGG-16 和 ResNet。这符合分辨率降低一半后通道数加倍的传统。

The CIFAR, SVHN, and ImageNet datasets serve as testbeds for comparison to prior work and analysis of FractalNet's internal behavior. We evaluate performance on the standard classification task associated with each dataset. For CIFAR and SVHN, which consist of 32×32 images, we set our fractal network to have 5 blocks ($B = 5$) with 2×2 non-overlapping max-pooling and subsampling applied after each. This reduces the input 32×32 spatial resolution to 1×1 over the course of the entire network. A softmax prediction layer attaches at the end of the network. Unless otherwise noted, we set the number of filter channels within blocks 1 through 5 as $(64, 128, 256, 512, 512)$, mostly matching the convention of doubling the number of channels after halving spatial resolution.

For ImageNet, we choose a fractal architecture to facilitate direct comparison with the 34-layer ResNet of He et al. (2016a). We use the same first and last layer as ResNet-34, but change the middle of the network to consist of 4 blocks ($B = 4$), each of 8 layers ($C = 4$ columns). We use a filter channel progression of $(128, 256, 512, 1024)$ in blocks 1 through 4.

4.1 Training

For experiments using dropout, we fix drop rate per block at $(0\%, 10\%, 20\%, 30\%, 40\%)$, similar to Clevert et al. (2016). Local drop-path uses 15% drop rate across the entire network.

[1]Densely connected networks (DenseNets) are concurrent work, appearing subsequent to our original arXiv paper on FractalNet. A variant of residual networks, they swap addition for concatenation in the residual functional form. We report performance of their 250-layer DenseNet-BC network with growth rate $k = 24$.

[2]This deeper (4 column) FractalNet has fewer parameters. We vary column width: $(128, 64, 32, 16)$ channels across columns initially, doubling each block except the last. A linear projection temporarily widens thinner columns before joins. As in Iandola et al. (2016), we switch to a mix of 1×1 and 3×3 convolutional filters.

Method	Top-1 (%)	Top-5 (%)
VGG-16	28.07	9.33
ResNet-34 C	24.19	7.40
FractalNet-34	24.12	7.39

Table 2: **ImageNet** (validation set, 10-crop).

Cols.	Depth	Params.	Error (%)
1	5	0.3M	37.32
2	10	0.8M	30.71
3	20	2.1M	27.69
4	40	4.8M	27.38
5	80	10.2M	26.46
6	160	21.1M	27.38

Table 3: **Ultra-deep fractal networks** (CIFAR-100++). Increasing depth greatly improves accuracy until eventual diminishing returns. Contrast with plain networks, which are not trainable if made too deep (Table 4).

表 3 表明 FractalNet 在通过增加 C 得到极深网络时性能并不下降 (C=6 对应 160 层)。这个表里的分数不能跟表 1 里的比。

Model	Depth	Train Loss	Error (%)
Plain	5	0.786	36.62
Plain	10	0.159	32.47
Plain	20	0.037	31.31
Plain	**40**	**0.580**	**38.84**
Fractal Col #1	5	0.677	37.23
Fractal Col #2	10	0.141	32.85
Fractal Col #3	20	0.029	31.31
Fractal Col #4	**40**	**0.016**	**31.75**
Fractal Full	**40**	**0.015**	**27.40**

Table 4: **Fractal structure as a training apparatus** (CIFAR-100++). Plain networks perform well if moderately deep, but exhibit worse convergence during training if instantiated with great depth. However, as a column trained within, and then extracted from, a fractal network with mixed drop-path, we recover a plain network that overcomes such depth limitation (possibly due to a student-teacher effect).

表 4 提供的基准显示，普通深度神经网络当层数达到 40 后就会下降。我们的经验中，普通的 160 层的网络完全不能收敛。表 3 还高亮了使用 FractalNet 以及路径舍弃作为提取已训练好网络的能力。

对于使用 dropout 的实验，我们固定每块的丢弃率为 (0%, 10%, 20%, 30%, 40%)。局部路径舍弃采用 15% 的舍弃率。

我们在 CIFAR-10/CIFAR-100 上训练了 400 个 epoch，在 SVHN 上训练了 20 个 epoch；每当剩余的 epoch 数减半，我们就把学习速率缩小 10。学习速率初始为 0.2。我们使用随机梯度下降方法，批处理大小为 100，冲量为 0.9。对参数使用 Xavier 初始化。

对 CIFAR-10 和 CIFAR-100 有标准的数据增强技术，仅包含平移和镜像翻转。平移量是 -4 到 4 的随机数。有必要的时候图像在减去平均值后以 0 补全。一半的图像水平被翻转。我们给使用的数据增强不超过这些所得到的结果标记为“ + ”。标记为“ ++ ”的使用了更多的增强技术；精确的策略可能会不一样。

We run for 400 epochs on CIFAR, 20 epochs on SVHN, and 70 epochs on ImageNet. Our learning rate starts at 0.02 (for ImageNet, 0.001) and we train using stochastic gradient descent with batch size 100 (for ImageNet, 32) and momentum 0.9. For CIFAR/SVHN, we drop the learning rate by a factor of 10 whenever the number of remaining epochs halves. For ImageNet, we drop by a factor of 10 at epochs 50 and 65. We use Xavier initialization (Glorot & Bengio, 2010).

A widely employed (Lin et al., 2013; Clevert et al., 2016; Srivastava et al., 2015; He et al., 2016a;b; Huang et al., 2016b; Targ et al., 2016) scheme for data augmentation on CIFAR consists of only horizontal mirroring and translation (uniform offsets in $[-4, 4]$), with images zero-padded where needed after mean subtraction. We denote results achieved using no more than this degree of augmentation by appending a "+" to the dataset name (*e.g.* CIFAR-100+). A "++" marks results reliant on more data augmentation; here exact schemes may vary. Our entry in this category is modest and simply changes the zero-padding to reflect-padding.

4.2 Results

Table 1 compares performance of FractalNet on CIFAR and SVHN with competing methods. FractalNet (depth 20) outperforms the original ResNet across the board. With data augmentation, our CIFAR-100 accuracy is close to that of the best ResNet variants. With neither augmentation nor regularization, FractalNet's performance on CIFAR is superior to both ResNet and ResNet with stochastic depth, suggesting that FractalNet may be less prone to overfitting. Most methods perform similarly on SVHN. Increasing depth to 40, while borrowing some parameter reduction tricks (Iandola et al., 2016), reveals FractalNet's performance to be consistent across a range of configuration choices.

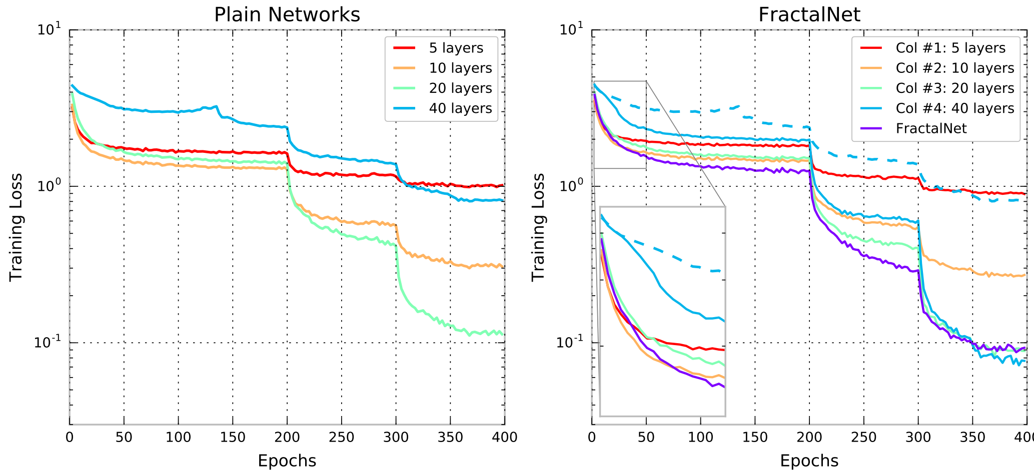
不使用数据增强的实验强调了路径舍弃正则化的能力。对于 CIFAR-100，添加路径舍弃把错误率从 35.34% 降低到 28.20%。未正则化的 ResNet 则差得远 (44.76%)，而通过随机深度正则化的 ResNet 达到的错误率为 37.80%，还不如我们未正则化的错误率 35.34%。CIFAR-10 上表现类似。

Experiments without data augmentation highlight the power of drop-path regularization. On CIFAR-100, drop-path reduces FractalNet's error rate from 35.34% to 28.20%. Unregularized ResNet is far behind (44.76%) and ResNet with stochastic depth (37.80%) does not catch up to our unregularized starting point of 35.34%. CIFAR-10 mirrors this story. With data augmentation, drop-path provides a boost (CIFAR-10), or does not significantly influence FractalNet's performance (CIFAR-100).

与数据增强组合起来后，路径舍弃要么提高了精度，或者效果不显著。通过路径舍弃，FractalNet 在 CIFAR-10 上达到所有模型中的最佳误差率。

Note that the performance of the deepest column of the fractal network is close to that of the full network (statistically equivalent on CIFAR-10). This suggests that the fractal structure may be more important as a learning framework than as a final model architecture.

Table 2 shows that FractalNet scales to ImageNet, matching ResNet (He et al., 2016a) at equal depth. Note that, concurrent with our work, refinements to the residual network paradigm further improve the state-of-the-art on ImageNet. Wide residual networks (Zagoruyko & Komodakis, 2016) of 34-layers reduce single-crop Top-1 and Top-5 validation error by approximately 2% and 1%, respectively, over



看图 3 中 40 层的分形网络训练期间的演化。通过追踪每一列我们发现，40 层的列一开始改进缓慢，但当网络中其它部分的损失开始稳定后会加快。普通的 40 层网络从来不会快速改进。

Figure 3: **Implicit deep supervision.** *Left*: Evolution of loss for plain networks of depth 5, 10, 20 and 40 trained on CIFAR-100. Training becomes increasingly difficult for deeper networks. At 40 layers, we are unable to train the network satisfactorily. *Right*: We train a 4 column fractal network with mixed drop-path, monitoring its loss as well as the losses of its four subnetworks corresponding to individual columns of the same depth as the plain networks. As the 20-layer subnetwork starts to stabilize, drop-path puts pressure on the 40-layer column to adapt, with the rest of the network as its teacher. This explains the elbow-shaped learning curve for Col #4 that occurs around 25 epochs.

ResNet-34 by doubling feature channels in each layer. DenseNets (Huang et al., 2016a) substantially improve performance by building residual blocks that concatenate rather than add feature channels.

Table 3 demonstrates that FractalNet resists performance degradation as we increase C to obtain extremely deep networks (160 layers for $C = 6$). Scores in this table are not comparable to those in Table 1. For time and memory efficiency, we reduced block-wise feature channels to $(16, 32, 64, 128, 128)$ and the batch size to 50 for the supporting experiments in Tables 3 and 4.

Table 4 provides a baseline showing that training of plain deep networks begins to degrade by the time their depth reaches 40 layers. In our experience, a plain 160-layer completely fails to converge. This table also highlights the ability to use FractalNet and drop-path as an engine for extracting trained networks (columns) with the same topology as plain networks, but much higher test performance.

4.3 Introspection

With Figure 3, we examine the evolution of a 40-layer FractalNet during training. Tracking columns individually (recording their losses when run as stand-alone networks), we observe that the 40-layer column initially improves slowly, but picks up once the loss of the rest of the network begins to stabilize. Contrast with a plain 40-layer network trained alone (dashed blue line), which never makes fast progress. The column has the same initial plateau, but subsequently improves after 25 epochs, producing a loss curve uncharacteristic of plain networks.

我们猜测分形结构触发了类似于深度监督和以及师徒信息流的效应。第 4 列每隔一层就与第 3 列连接，而每隔 4 层这个连接不涉及别的列。当分形网络部分依赖于信号从第 3 列流过，路径舍弃对第 4 列施加压力，以产生一个当第 3 列被移除时的替代信号。一个特定的舍弃仅仅需要第 4 列中两个相邻层代替第 3 列中的一层。这相当于小小的师徒问题。

We hypothesize that the fractal structure triggers effects akin to deep supervision and lateral student-teacher information flow. Column #4 joins with column #3 every other layer, and in every fourth layer this join involves no other columns. Once the fractal network partially relies on the signal going through column #3, drop-path puts pressure on column #4 to produce a replacement signal when column #3 is dropped. This task has constrained scope. A particular drop only requires two consecutive layers in column #4 to substitute for one in column #3 (a mini student-teacher problem).

This explanation of FractalNet dynamics parallels what, in concurrent work, Greff et al. (2017) claim for ResNet. Specifically, Greff et al. (2017) suggest residual networks learn unrolled iterative estimation, with each layer performing a gradual refinement on its input representation. The deepest FractalNet column could behave in the same manner, with the remainder of the network acting as a scaffold for building smaller refinement steps by doubling layers from one column to the next.

These interpretations appear not to mesh with the conclusions of Veit et al. (2016), who claim that ensemble-like behavior underlies the success of ResNet. This is certainly untrue of some very deep networks, as FractalNet provides a counterexample: we can extract a single column (plain network topology) and it alone (no ensembling) performs nearly as well as the entire network. Moreover, the gradual refinement view may offer an alternative explanation for the experiments of Veit et al. (2016). If each layer makes only a small modification, removing one may look, to the subsequent portion of the network, like injecting a small amount of input noise. Perhaps noise tolerance explains the gradual performance degradation that Veit et al. (2016) observe when removing ResNet layers.

5 Conclusion

FractalNet 展示了路径长度对训练极深神经网络至关重要；而残差的影响是偶然的。关键在于 FractalNet 和 ResNet 具有共同特性：标称网络深度很大，但训练过程中梯度传播的有效路径更短。分形结构可能是满足这一需求的最简单方式，并且能达到甚至超越循环网络的经验性能。它们能避免深度太大；过多的深度会使训练变慢，但不会增加准确度。

通过 drop-path 极深分形网络的正则化是直接而有效的。对于需要快速回应的应用，drop-path 能对分形网络里的延迟和精确性做折中。

我们的分析连接了分形网络中的突发内部行为与构建在其他设计中的现象。分形网络的子结构与一些卷积网络中作为结构模块的人工设计模块相似。它们的进化或许能仿真深度监视和师徒学习。

Our experiments with fractal networks provide strong evidence that path length is fundamental for training ultra-deep neural networks; residuals are incidental. Key is the shared characteristic of FractalNet and ResNet: large nominal network depth, but effectively shorter paths for gradient propagation during training. Fractal architectures are arguably the simplest means of satisfying this requirement, and match residual networks in experimental performance. Fractal networks are resistant to being too deep; extra depth may slow training, but does not impair accuracy.

With drop-path, regularization of extremely deep fractal networks is intuitive and effective. Drop-path doubles as a method of enforcing speed (latency) vs. accuracy tradeoffs. For applications where fast responses have utility, we can obtain fractal networks whose partial evaluation yields good answers.

Our analysis connects the internal behavior of fractal networks with phenomena engineered into other networks. Their substructure resembles hand-crafted modules used as components in prior work. Their training evolution may emulate deep supervision and student-teacher learning.

Acknowledgments

We gratefully acknowledge the support of NVIDIA Corporation with the donation of GPUs used for this research. This work was partially supported by the NSF award RI:1409837.

References

Jimmy Ba and Rich Caruana. Do deep nets really need to be deep? *NIPS*, 2014.

Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. Fast and accurate deep network learning by exponential linear units (ELUs). *ICLR*, 2016.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. *CVPR*, 2009.

Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. *AISTATS*, 2010.

Benjamin Graham. Fractional max-pooling. *arXiv:1412.6071*, 2014.

Klaus Greff, Rupesh Kumar Srivastava, and Jürgen Schmidhuber. Highway and residual networks learn unrolled iterative estimation. *ICLR*, 2017.

Bharath Hariharan, Pablo Arbelaez, Ross Girshick, and Jitendra Malik. Hypercolumns for object segmentation and fine-grained localization. *CVPR*, 2015.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification. *ICCV*, 2015.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CVPR*, 2016a.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. *ECCV*, 2016b.

Geoffrey E. Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv:1207.0580*, 2012.

Gao Huang, Zhuang Liu, and Kilian Q. Weinberger. Densely connected convolutional networks. *arXiv:1608.06993*, 2016a.

Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Weinberger. Deep networks with stochastic depth. *ECCV*, 2016b.

Forrest N. Iandola, Matthew W. Moskewicz, Khalid Ashraf, Song Han, William J. Dally, and Kurt Keutzer. SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <1MB model size. *arXiv:1602.07360*, 2016.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *ICML*, 2015.

Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv:1408.5093*, 2014.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, 2009.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. ImageNet classification with deep convolutional neural networks. *NIPS*, 2012.

Quoc V Le, Navdeep Jaitly, and Geoffrey E Hinton. A simple way to initialize recurrent networks of rectified linear units. *arXiv:1504.00941*, 2015.

Chen-Yu Lee, Saining Xie, Patrick Gallagher, Zhengyou Zhang, and Zhuowen Tu. Deeply-supervised nets. *NIPS Workshop on Deep Learning and Representation Learning*, 2014.

Chen-Yu Lee, Patrick W Gallagher, and Zhuowen Tu. Generalizing pooling functions in convolutional neural networks: Mixed, gated, and tree. *AISTATS*, 2016.

Ming Liang and Xiaolin Hu. Recurrent convolutional neural network for object recognition. *CVPR*, 2015.

Zhibin Liao and Gustavo Carneiro. Competitive multi-scale convolution. *arXiv:1511.05635*, 2015.

Min Lin, Qiang Chen, and Shuicheng Yan. Network in network. *ICLR*, 2013.

Michael Maire, Stella X. Yu, and Pietro Perona. Reconstructive sparse code transfer for contour detection and semantic labeling. *ACCV*, 2014.

Dmytro Mishkin and Jiri Matas. All you need is a good init. *ICLR*, 2016.

Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. *ICML*, 2010.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng. Reading digits in natural images with unsupervised feature learning. *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011.

Behnam Neyshabur, Ruslan Salakhutdinov, and Nathan Srebro. Path-SGD: Path-normalized optimization in deep neural networks. *NIPS*, 2015.

Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. Fitnets: Hints for thin deep nets. *ICLR*, 2015.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *ICLR*, 2015.

Jasper Snoek, Oren Rippel, Kevin Swersky, Ryan Kiros, Nadathur Satish, Narayanan Sundaram, Md Patwary, Mostofa Ali, Ryan P Adams, et al. Scalable bayesian optimization using deep neural networks. *ICML*, 2015.

Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller. Striving for simplicity: The all convolutional net. *ICLR (workshop track)*, 2014.

Rupesh Kumar Srivastava, Klaus Greff, and Jürgen Schmidhuber. Highway networks. *ICML*, 2015.

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. *CVPR*, 2015.

Sasha Targ, Diogo Almeida, and Kevin Lyman. Resnet in resnet: Generalizing residual architectures. *arXiv:1603.08029*, 2016.

Gregor Urban, Krzysztof J. Geras, Samira Ebrahimi Kahou, Ozlem Aslan, Shengjie Wang, Abdelrahman Mohamed, Matthai Philipose, Matt Richardson, and Rich Caruana. Do deep convolutional nets really need to be deep and convolutional? *ICLR*, 2017.

Andreas Veit, Michael Wilber, and Serge Belongie. Residual networks behave like ensembles of relatively shallow networks. *NIPS*, 2016.

Li Wan, Matthew Zeiler, Sixin Zhang, Yann L Cun, and Rob Fergus. Regularization of neural networks using dropconnect. *ICML*, 2013.

Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *BMVC*, 2016.