- Insights

Headline

大家好：

这是2018年度第12篇Arxiv Weekly。

本文是 **训练技巧** 方向的文章。

Highlight

> **本文综述了一系列利用正则化和增强，在cifar/imagenet上对抗overfitting的技术，其中state-of-the-art的部分对一些backbone的涨点较有实用价值**

Information

title

- A serial of works fight against overfitting

links

- paper list provided later

source

- Momenta Rocker

Preview of Overfitting

成因

由于模型空间远远大于数据空间。也即参数过度冗余，训练数据不足，独立方程数目远少于未知数的数目，因此求出来的未知数可以有很多组解。

解决手段

- **enlarge training dataset**
  原理是显然的，但是有烧钱的问题。而且随着dataset的增长，计算等资源也会同步加快消耗。不能作为解决问题的唯一手段。
- **cross validation & early stop**
  不能涨点，只能准确预测模型性能，选出最好模型并加快迭代。
- **regularization related**
  正则化是防止overfitting的核心技术，通过对训练引入某种先验信息约束防止拟合走向错误的极端。
  - *weight decay*：在正常的loss后面附加参数的范数项，通过假定模型参数具有高斯/拉布拉斯分布，使得参数空间极大压缩，许多奇异的overfit点就被抹去了。
  - *normalization*：一种如今看来十分令人无奈的技术，Hinton的心理阴影。对网络的训练有至关重要而难以完美解释的作用。实际上normalization也存在一系列的技术，BN；LN；GN；IBN；SN等，可以专门分享一个

小时。

- **data related**
  许多技术对数据进行操作，期望获得更合理的分布或者变相增加数据集。
  - *data augmentation*：变相增大数据集。
  - *data shuffle*：常规操作，通过数据集的shuffle方式输入高度有序的训练样本后，模型对输入数据顺序进行拟合（一种特殊的过拟合）
  - *label smoothing*：为了防止label的over confidence，将gt中true的confidence改为0.9，false的confidence改为0.1。

Regularization

1.Dropout & DropConnect

paperlink

- **dropout:**
  G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. CoRR, abs/1207.0580, 2012.
- **dropconnect:**
  Li Wan, Matthew Zeiler, Sixin Zhang, Yann L Cun, and Rob Fergus. Regularization of neural networks using dropconnect. ICML, 2013.

keys

dropout是这个领域最早的工作，核心思想是根据给定的超参数dropout rate，在训练时将FC层的输出进行elementwise的随机置零。

dropout压制了参数的过冗余，并使网络能够学习到更鲁棒、更高级的信息，而不是对无意义信息过拟合。

也有人认为，dropout本质上是一种ensemble技术，每个模型都基于前面模型的基础train一次，最终的模型是由一系列共享参数的模型演化生成。

$$y = mask \odot Activate(Wx)$$

而dropconnect是对dropout的自然扩展，直接对FC层的参数W进行elementwise的随机置零，等价于对FC层前后神经元的connection进行dropping操作，也因此得名。

$$y = Activate((mask \odot W)x)$$

results

在Alexnet等早期网络上，dropout有奇效，而dropconnect能够取得比dropout优化一些的点数。

model	error(%)
No-Drop	23.5
Dropout	19.7
DropConnect	**18.7**

Table 4. CIFAR-10 classification error using the simple feature extractor described in (Krizhevsky, 2012)(layers-80sec.cfg) and with no data augmentation.

2.Maxout

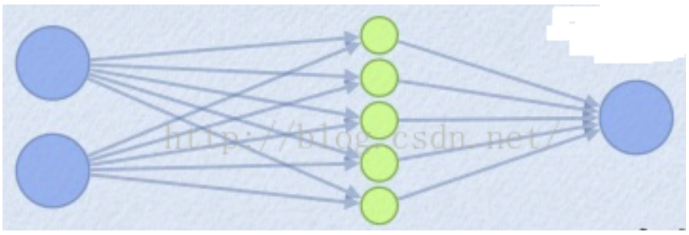
paperlink

- Goodfellow I J, Warde-Farley D, Mirza M, et al. Maxout networks[J]. arXiv preprint arXiv:1302.4389, 2013.

keys

maxout是13年Goodfellow在小数据集上state-of-the-art的工作。

maxout的操作十分朴素，对原本的FC层进行K倍增操作，然后从倍增后得到的K个输出中选择一个最大的作为final output。



实际上maxout是一种可学习的激活函数，如果W的规模足够大，K=2的maxout就能够拟合任意的连续函数。

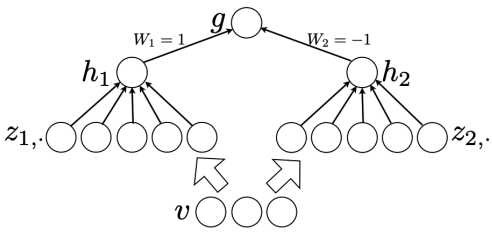


Figure 3. An MLP containing two maxout units can arbitrarily approximate any continuous function. The weights in the final layer can set g to be the difference of h_1 and h_2. If z_1 and z_2 are allowed to have arbitrarily high cardinality, h_1 and h_2 can approximate any convex function. g can thus approximate any continuous function due to being a difference of approximations of arbitrary convex functions.

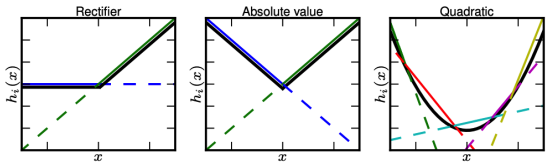


Figure 1. Graphical depiction of how the maxout activation function can implement the rectified linear, absolute value rectifier, and approximate the quadratic activation function. This diagram is 2D and only shows how maxout behaves with a 1D input, but in multiple dimensions a maxout unit can approximate arbitrary convex functions.

可能是由于这样的激活操作过于consuming，而且后期网络规模越来越大，对每个部件的效率要求越来越高，目前似乎maxout操作已经没什么人用

3. Spatial Dropout

paperlink

- Jonathan Tompson, Ross Goroshin, Arjun Jain, Yann LeCun, and Christoph Bregler. Efficient object localization using convolutional networks. In CVPR, 2015.

keys

显然dropout机制本身无法直接很好地作用于卷积层。其中主要有两个原因。

1. 卷积层的冗余度没有FC层那么大，尤其对于前期卷积层暴力添加dropout有导致欠拟合掉点的风险，超参数不好控制。
2. 卷积层的输出是高度耦合相关的，一个信息可能包含在一个区域当中。这样的耦合使得单纯添加传统的dropout不能实现去除网络冗余表达能力的意图。(这样的相关性主要存在于同一个feature map的相邻pixel间)

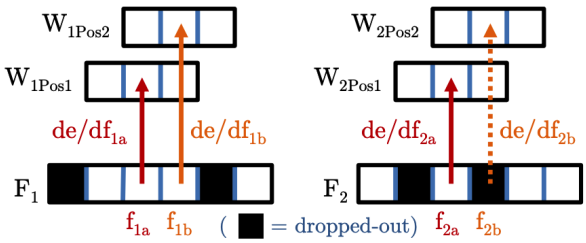


Figure 3: Standard Dropout after a 1D convolution layer

因此一个很容易想到的方法是，对于后期的已经高度抽象的卷积层，进行channelwise的随机置零，也即spatial dropout。

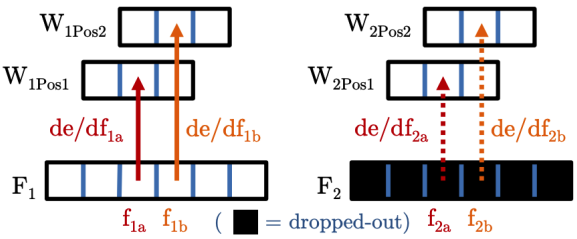


Figure 5: *SpatialDropout* after a 1D convolution layer

本文中核心问题是检测，spatial dropout作为一个技术细节被提及，具体的result不是本文的重点。

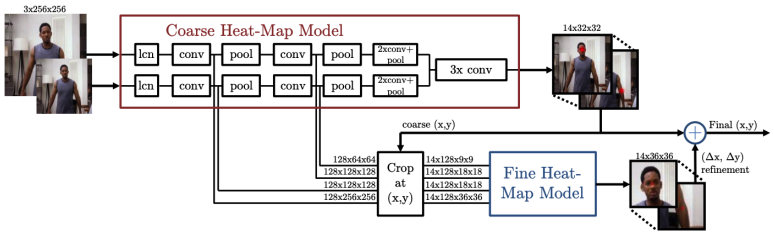


Figure 4: Overview of our Cascaded Architecture

4. DropPath

paperlink

- Larsson G, Maire M, Shakhnarovich G. Fractalnet: Ultra-deep neural networks without residuals[J]. arXiv preprint arXiv:1605.07648, 2016.

keys

实际上，随着inception和resnet的大热，许多存在skip connection、multi path的网络结构被设计出来。

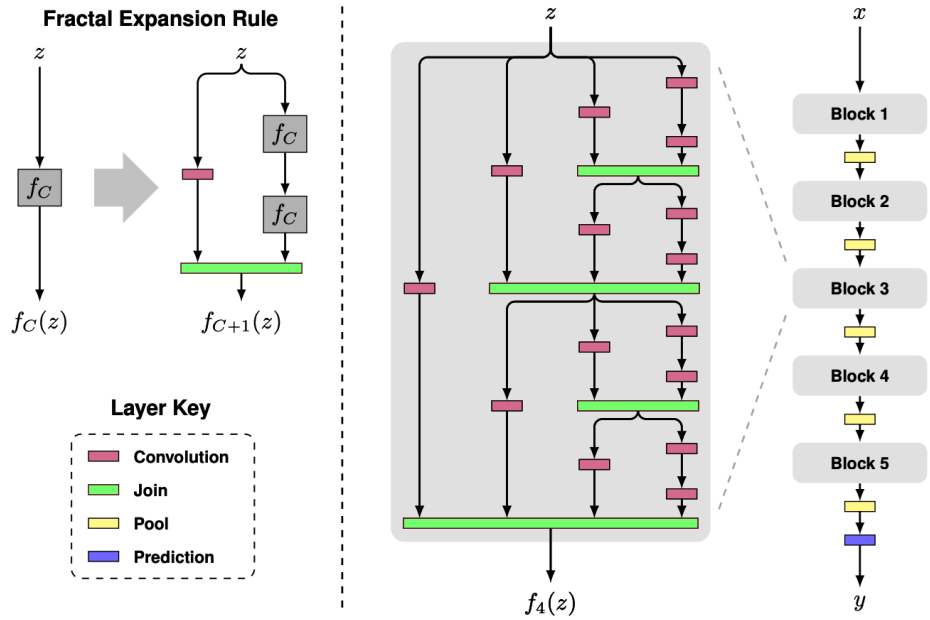


Figure 1: **Fractal architecture.** *Left:* A simple expansion rule generates a fractal architecture with C intertwined columns. The base case, $f_1(z)$, has a single layer of the chosen type (*e.g.* convolutional) between input and output. Join layers compute element-wise mean. *Right:* Deep convolutional 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 ($B = 5$, $C = 4$).

这意味着，原本作用于FC的dropout思想又多了一个可以施展的维度：将multi path network中的路径以一定的概率drop掉，即为droppath。

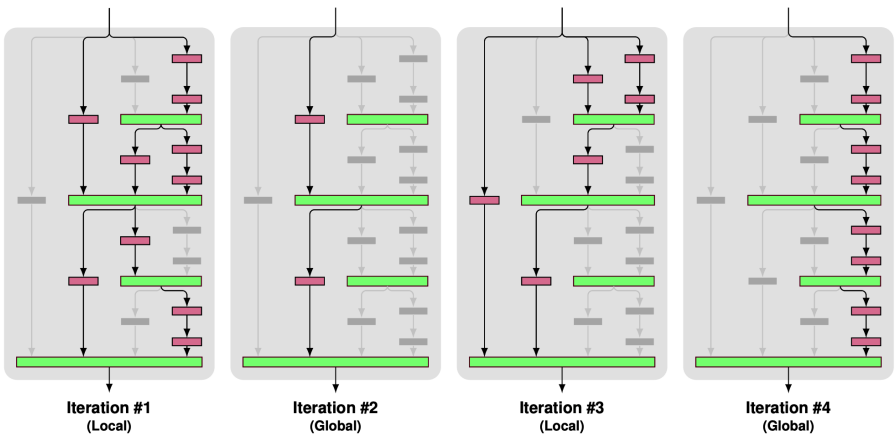


Figure 2: **Drop-path.** A fractal network block functions with some connections between layers disabled, provided some path from input to output is still available. Drop-path guarantees at least one such path, while sampling a subnetwork with many other paths disabled. During training, presenting a 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.

result

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.

可以看到对于作者提出的复杂多路并行网络，dropath是一种makesense的正则方式，在许多情况下能够涨点。

5. ScheduledDropPath

paperlink

- Zoph B, Vasudevan V, Shlens J, et al. Learning transferable architectures for scalable image recognition[J]. arXiv preprint arXiv:1707.07012, 2017, 2(6).

keys

本文为google brain在NAS方向上的工作。文章在实验中发现了一件有趣的事情，也即利用DropPath本身无法给NasNet带来任何好处；但是如果随着训练逐渐增加drop probability，则能够极大提升NasNet的performance。作者将这种trick命名为ScheduledDropPath

6. StochasticDepth (RandomDrop)

paperlink

- Huang G, Sun Y, Liu Z, et al. Deep networks with stochastic depth[C] European Conference on Computer Vision. Springer, Cham, 2016: 646-661.

keys

显然上面的两种思路只能适用于含有inception结构的网络或者是ResNeXt结构的网络。对于纯粹的ResNet并不实用。

故而基本同时，应用于超深（超过1000层）的ResNet网络的dropping思路被提出，也即StochasticDepth，随机深度网络。

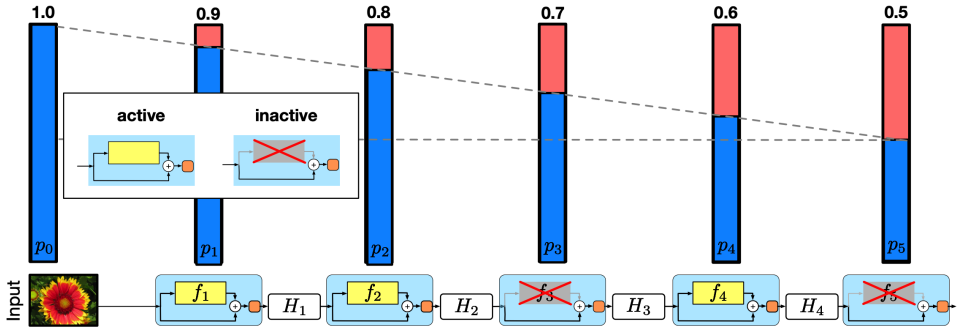


Fig. 2. The linear decay of p_ℓ illustrated on a ResNet with stochastic depth for $p_0 = 1$ and $p_L = 0.5$. Conceptually, we treat the input to the first ResBlock as H_0, which is always active.

核心的想法非常直白，就是对于极深的ResNet网络，在训练的时候所有的Residual Block被以一定的概率drop掉；而在推理的时候则保持网络的完整性，只是乘以本层drop probability的均值以校准网络。

于是被random drop的网络就相当于有了stochastic depth（因此这个工作一般用这两个名字简称），从而给训练过程中的特征抽象引入随机性，进一步对抗参数量巨大的深度网络overfitting的问题。

本文也采用了线性变化的drop rate，相当于是一种“空间域的schedule”，重要的浅层block以更大概率保留，越深层的block被drop的概率越大。

$$P_l = 1 - \frac{l}{L}(1 - P_L)$$

其中 l 为resdual block的index；L为总block数；P_l为第 l 个block的residual部分被drop掉的概率；P_L为给定的drop rate，是超参数。

results

重要的参数：$P_L = 0.5$，cifar中使用Kaiming原文中的ResNet110结构。

Table 1. Test error (%) of ResNets trained with stochastic depth compared to other most competitive methods previously published (whenever available). A "+" in the name denotes standard data augmentation. ResNet with constant depth refers to our reproduction of the experiments by He et al.

	CIFAR10+	CIFAR100+	SVHN	ImageNet
Maxout [21]	9.38	-	2.47	-
DropConnect [20]	9.32	-	1.94	-
Net in Net [24]	8.81	-	2.35	-
Deeply Supervised [13]	7.97	-	1.92	33.70
Frac. Pool [25]	-	27.62	-	-
All-CNN [6]	7.25	-	-	41.20
Learning Activation [26]	7.51	30.83	-	-
R-CNN [27]	7.09	-	1.77	-
Scalable BO [28]	6.37	27.40	1.77	-
Highway Network [29]	7.60	32.24	-	-
Gen. Pool [30]	6.05	-	1.69	28.02
ResNet with constant depth	6.41	27.76	1.80	21.78
ResNet with stochastic depth	5.25	24.98	1.75	21.98

可以看到，stochastic depth确实有不错的涨点，达到了当时的state-of-the-art结果。比较重要的是，对于深度的resnet网络这是一种通用的白捞好处的正则化手段。

7. Shake-Shake

paperlink

- Gastaldi X. Shake-shake regularization[J]. arXiv preprint arXiv:1705.07485, 2017.

keys

shake-shake是个奇异的名字（不得不让我们想起百度的paddle-paddle），不过不可否认它的出现代表了cifar的new state-of-the-art。

shake-shake做了一件之前很少有人做过的事情，即在前传和反传引入不同的随机性。作者将这波奇异的操作命名为颤抖（shake），大概是设想到一个前传反传包含不同随机性的网络，在训练中会不断的shake-shake and shake again。

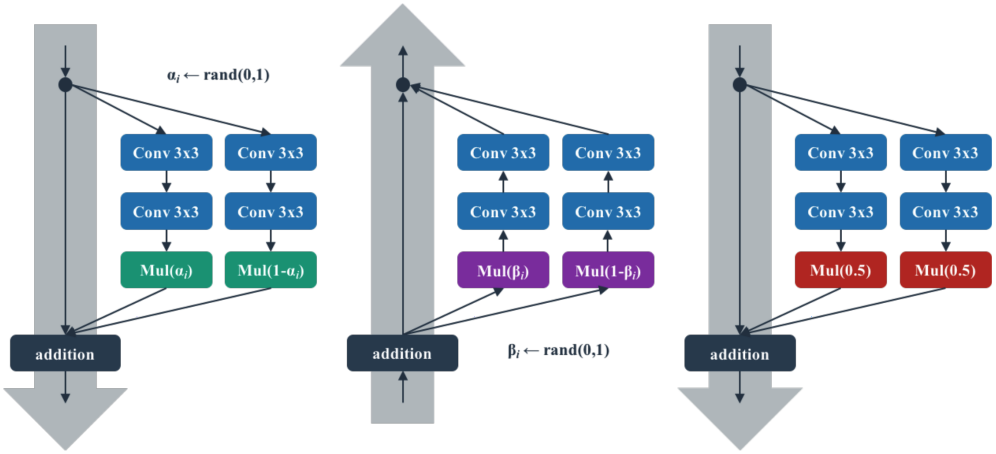


Figure 1: **Left:** Forward training pass. **Center:** Backward training pass. **Right:** At test time.

具体来说，shake-shake必须应用在类似resNeXt的结构上，也即存在多路同构的residual path。

训练过程中，在前传的时候两路residual path会被随机加权后合并，且权重的和为1；反传的时候也一样，但是加权用的随机变量是重新生成的。

在推理的时候，多路同构的residual path会同权重接入网络，且权重为随机权重变量的均值（为了校准网络）。

results

Table 1: Error rates (%) on CIFAR-10. Results that surpass all competing methods by more than 0.1% are **bold** and the overall best result is **blue**.

			Model		
Forward	Backward	Level	26 2x32d	26 2x64d	26 2x96d
Even	Even	n/a	4.27	3.76	3.58
Even	Shake	Batch	4.44	-	-
Shake	Keep	Batch	4.11	-	-
Shake	Even	Batch	3.47	**3.30**	-
Shake	Shake	Batch	3.67	**3.07**	-
Even	Shake	Image	4.11	-	-
Shake	Keep	Image	4.09	-	-
Shake	Even	Image	3.47	**3.20**	-
Shake	Shake	Image	3.55	**2.98**	**2.86**

对任意的training pass(forward / backward)

- shake 代表独立随机采样
- keep 代表采用和另一个方向相同的权重
- even 代表不采样，平均分配权重

另外：

- Image 代表对每个数据独立采样
- Batch 代表对整个batch仅采样一次，数据间公用权重

可见，cifar10上最佳的方案为shake-shake-image，这也是本文名称的由来。

shake-shake主要是一种正则化的方式，但是也提出了自己独立的backbone，故而可以和其他的backbone的性能进行对比。可以看到在点数上确实是state-of-the-art的。而且所需要的网络层数只有26层，可见并不是靠无脑增大网络实现突破。

Table 3: Test error (%) and model size on CIFAR. Best results are **blue**.

Method	Depth	Params	C10	C100
Wide ResNet	28	36.5M	3.8	18.3
ResNeXt-29, 16x64d	29	68.1M	3.58	17.31
DenseNet-BC (k=40)	190	25.6M	3.46	17.18
C10 Model S-S-I	26	26.2M	**2.86**	-
C100 Model S-E-I	29	34.4M	-	**15.85**

8. ShakeDrop

paperlink

- Yamada Y, Iwamura M, Kise K. ShakeDrop regularization[J]. arXiv preprint arXiv:1802.02375, 2018.
- DeVries T, Taylor G W. Dataset augmentation in feature space[J]. arXiv preprint arXiv:1702.05538, 2017.

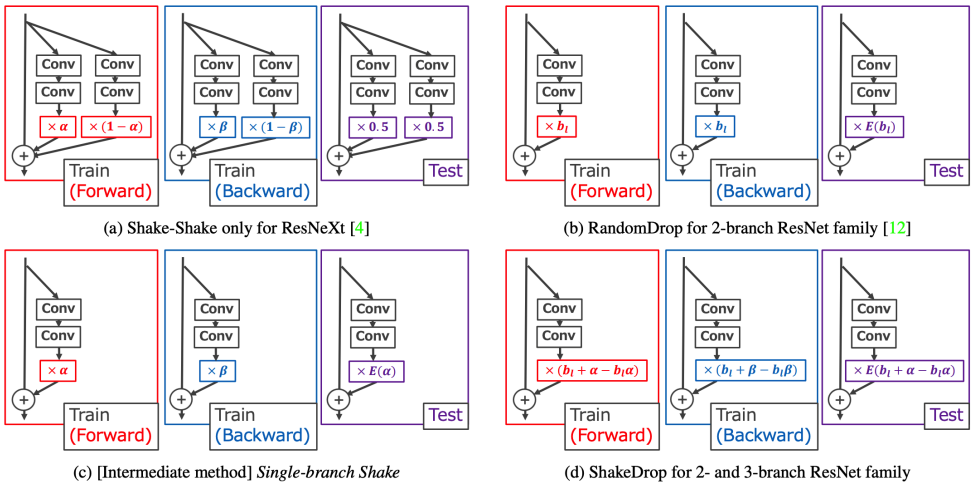
keys

shake-drop解决了如何将shake-shake的训练方式运用在更common的backbone的问题。

首先说明，如果强行将shake-shake中的方法用在single branch上，会导致训练的崩溃。希望运用这样的方法首先需要分析shake-shake work的原因和single branch shake-shake崩溃的原因。

> 今年ICLR workshop中的文章 *Dataset augmentation in feature space* 提出可以通过对网络中feature进行插值或者噪声附加，实现全网络的数据增强。
>
> 而本文提出，shake-shake之所以work，可以理解为 **前传的时候通过随机权重合并两个同构支路的feature，在网络的每一个地方都完成了恰如其分的增强**。
>
> 而 **反传的时候的随机权重本质上是一种扰乱**，使得前传时候两个同构支路的随机合并有意义。（试想如果正常反传，随着训练进行两个支路理论上是完全趋同的，就达不到增强的效果）
>
> （实际上如果网络能够抽象图像的根本性特征，那么同一张图无论怎么引入两个支路上分布的随机性，这些共同的根本性特征是不变的。shake-shake其实利用了这一点，在网络的各个层次进行增强操作，从而一举突破到state-of-the-art。）
>
> 以上特性使我们需要保留的，下面需要知道single branch shake-shake崩溃的原因以避免之
>
> 显然在shake-shake的训练过程中，相比于正常的网络，反传的梯度乘上了如下的常数。
>
> $$\frac{\beta}{\alpha};\frac{1-\beta}{1-\alpha}$$
>
> **如果α十分靠近0或者1，则上两个式子会有一个奇异。这时候另外一个还是完好的，能一定程度挽回性能的损失。**
>
> 显然single branch shake-shake就不存在这样的机制。故而会导致bug of stabilization

综上，文章作者给出了自己的single branch shake-shake方案，也即所谓的shake drop



(a) Shake-Shake only for ResNeXt [4]

(b) RandomDrop for 2-branch ResNet family [12]

(c) [Intermediate method] *Single-branch Shake*

(d) ShakeDrop for 2- and 3-branch ResNet family

Figure 1. Regularization methods for ResNet family. Conv represents convolution layer, $E[x]$ expected value of x, and α, β and b_l random coefficients which are changed on every iteration. (a) and (b) are existing methods. (c) is an intermediate regularization method *Single-branch Shake* to the proposed method. (d) is the proposed method.

从上图能非常清楚地看到四种方式的对比

$$G(x) = \begin{cases} x + (b_l + \alpha - b_l\alpha)F(x), & \text{in train-fwd} \\ x + (b_l + \beta - b_l\beta)F(x), & \text{in train-bwd} \\ x + E[b_l + \alpha - b_l\alpha]F(x), & \text{in test,} \end{cases} \quad (6)$$

上式为shake drop的迭代公式，可以看到，这个公式是shake-shake和random drop的结合，且在极限情况下：

In the training phase, b_l controls the behaviour of ShakeDrop. If $b_l = 1$, Eqn. (6) is deformed as

$$G(x) = \begin{cases} x + F(x), & \text{in train-fwd} \\ x + F(x), & \text{in train-bwd.} \end{cases} \quad (7)$$

That is, ShakeDrop is equivalent to the original network (e.g., ResNet). If $b_l = 0$, Eqn. (6) is deformed as

$$G(x) = \begin{cases} x + \alpha F(x), & \text{in train-fwd} \\ x + \beta F(x), & \text{in train-bwd.} \end{cases} \quad (8)$$

That is, calculation of $F(x)$ is perturbated by α and β.

因此，shake-drop解决上述问题的手段如下：

1. 通过在residual path上附加噪声实现shaking augment，于是摆脱了必须存在同构分枝的约束。
2. 引用random drop的思想，赋予unstable的反传一个概率，使得整体网络变为shake-shake和vanilla的加权和，其中权重（即概率）是一个超参。这样结构的stabilization就大大提高了。

results

在cifar10上结果如下

Table B1. Top-1 errors (%) at the final epoch (300th or 1800th) on the CIFAR-10/100 datasets. Representative methods and the proposed ShakeDrop applied to ResNet, ResNet (EraseReLU), and PyramidNet are compared. In ShakeDrop, $\alpha = 0, \beta \in [0, 1]$ were used in the original ResNet and $\alpha \in [-1, 1], \beta \in [0, 1]$ in the PyramidNet and the "EraseReLU"ed ResNet. "Gain" represents reduced error rates by longer training (i.e., Error rates of "300 Epoch" − Error rates of "1800 Epoch"). Positive value means longer training is effective. $*$ indicates the result is quoted from [4]. $+$ indicates the average result of 4 runs.

(a) CIFAR-10

Method	Regularization	Depth	#Param	300 Epoch	1800 Epoch	Gain
ResNeXt [27, 4]	-	26	26.2M	4.02	*3.58	0.44
	Shake-Shake	26	26.2M	3.36	*2.86	0.50
ResNet [11]	ShakeDrop	110	1.7M	4.78	**4.08**	0.70
ResNet (EraseReLU) [3]	ShakeDrop	110	1.7M	5.08	**3.81**	1.27
PyramidNet [9]	ShakeDrop	272	26.0M	3.41	**2.67**	0.74

(b) CIFAR-100

Method	Regularization	Depth	#Param	300 Epoch	1800 Epoch	Gain
ResNeXt [27, 4]	-	29	34.4M	16.95	*16.34	0.61
	Shake-Shake	29	34.4M	16.46	*15.85	0.61
ResNet [11]	ShakeDrop	110	1.7M	+23.74	**21.18**	2.56
ResNet (EraseReLU) [3]	ShakeDrop	110	1.7M	+21.81	**19.81**	2.00
PyramidNet [9]	ShakeDrop	272	26.0M	14.90	**13.99**	0.91

这里通过PyramidNet+shakedrop获取了2.67的state-of-the-art结果，但是其实结果的可比较性比较差。

在ImageNet上结果如下

Table B2. Top-1 errors (%) on ImageNet dataset at 90th and 180th epochs. This "Type" indicates either the original or "EraseReLU"ed networks. In ShakeDrop, $\alpha = 0, \beta \in [0, 1]$ were used in the original ResNet and the original ResNeXt, and $\alpha \in [-1, 1], \beta \in [0, 1]$ in the original PyramidNet and the "EraseReLU"ed networks. "Gain" represents reduced error rates by longer training (i.e., Error rates of "90 Epoch" − Error rates of "180 Epoch"). Positive value means longer training is effective.

Methods	Type	Regularization	p_L	90 Epoch	180 Epoch	Gain
ResNet-50 <Conv-BN-ReLU-Conv-BN-ReLU-Conv-BN-add>	Original	Vanilla	-	23.60	23.20	0.40
		RandomDrop	0.05	23.36	22.66	0.70
		ShakeDrop	0.05	23.33	**22.52**	**0.81**
	EraseReLU	Vanilla	-	25.63	25.03	0.60
		RandomDrop	0.05	25.61	24.77	0.84
		ShakeDrop	0.05	25.69	**24.53**	**1.16**
ResNeXt-50 32-4d <Conv-BN-ReLU-Conv-BN-ReLU-Conv-BN-add>	Original	Vanilla	-	22.78	22.83	-0.05
		RandomDrop	0.05	22.47	22.68	-0.21
		ShakeDrop	0.05	22.38	**22.34**	**0.04**
	EraseReLU	Vanilla	-	24.66	24.33	0.33
		RandomDrop	0.05	24.45	23.97	0.48
		ShakeDrop	0.05	24.41	**23.64**	**0.77**
PyramidNet-50 α300 <BN-Conv-BN-ReLU-Conv-BN-ReLU-Conv-BN-add>	Original	Vanilla	-	24.06	23.73	0.33
		RandomDrop	0.05	23.89	**23.24**	**0.65**
		ShakeDrop	0.05	23.95	23.33	0.62

可以看到基本也能够取得一定的好处。总结来看shake-drop当中提到了很多有意义的分析，对更深入理解shake-shake很有帮助。另外也找到了在通用网络中使用shake-shake策略的方式。

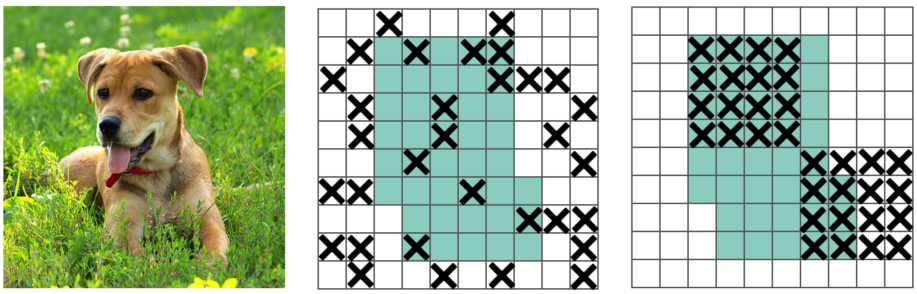
9. Dropblock

paperlink

- Ghiasi G, Lin T Y, Le Q V. DropBlock: A regularization method for convolutional networks[C]//Advances in Neural Information Processing Systems. 2018: 10748-10758.

keys

dropblock为另一条支线上的成果。继承了dropout-spatial dropout的脉络，可以视为dropout在卷积层上的泛化和推广，整体思路非常简单。

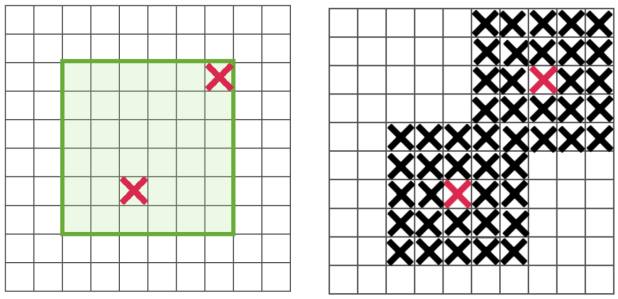


(a) (b) (c)

Figure 1: (a) input image to a convolutional neural network. The green regions in (b) and (c) include the activation units which contain semantic information in the input image. Dropping out activations at random is not effective in removing semantic information because nearby activations contain closely related information. Instead, dropping continuous regions can remove certain semantic information (e.g., head or feet) and consequently enforcing remaining units to learn features for classifying input image.

由于前述原因，对卷积层需要考虑整个feature map上的块状/整体dropping才有意义。

dropblock用如下机制生成一个随机的块状drop area，从而在通用的卷积神经网络中make sense地插入dropping操作。



(a) (b)

Figure 2: Mask sampling in DropBlock. (a) On every feature map, similar to dropout, we first sample a mask M. We only sample mask from shaded green region in which each sampled entry can expanded to a mask fully contained inside the feature map. (b) Every zero entry on M is expanded to $block\_size \times block\_size$ zero block.

results

Model	top-1(%)	top-5(%)
ResNet-50	76.51 ± 0.07	93.20 ± 0.05
ResNet-50 + dropout (kp=0.7) [1]	76.80 ± 0.04	93.41 ± 0.04
ResNet-50 + DropPath (kp=0.9) [17]	77.10 ± 0.08	93.50 ± 0.05
ResNet-50 + SpatialDropout (kp=0.9) [20]	77.41 ± 0.04	93.74 ± 0.02
ResNet-50 + Cutout [23]	76.52 ± 0.07	93.21 ± 0.04
ResNet-50 + AutoAugment [27]	77.63	93.82
ResNet-50 + label smoothing (0.1) [28]	77.17 ±0.05	93.45 ±0.03
ResNet-50 + DropBlock, (kp=0.9)	78.13 ± 0.05	94.02 ± 0.02
ResNet-50 + DropBlock (kp=0.9) + label smoothing (0.1)	78.35 ± 0.05	94.15 ± 0.03

Table 1: Summary of validation accuracy on ImageNet dataset for ResNet-50 architecture. For dropout, DropPath, and SpatialDropout, we trained models with different $keep\_prob$ values and reported the best result. DropBlock is applied with $block\_size = 7$. We report average over 3 runs.

可以看到从文中的实验结果来看，dropblock用于resnet有十分好的效果。事实上从复现结果看，还没有得到一个稳定的产出方式，距离paper report的结果有一定差距。

Data Augment

1. Cutout

paperlink

- DeVries T, Taylor G W. Improved regularization of convolutional neural networks with cutout[J]. arXiv preprint arXiv:1708.04552, 2017.

keys

一张图就足以说明cutout做的所有事情

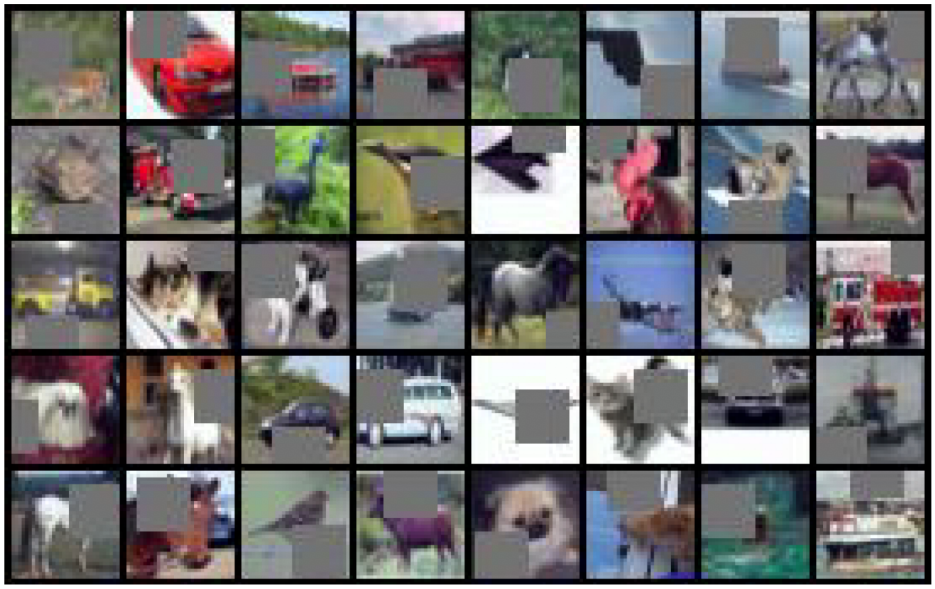


Figure 1: Cutout applied to images from the CIFAR-10 dataset.

cutout的insights来自检测中常遇到的遮挡问题，思路就是要打败敌人首先要了解敌人，人为生成大量含有恰当遮挡的图片交付给网络。这样真正遇到遮挡的时候网络会像打过疫苗一样，对这种情况产生很强的抗性。

令人惊喜的是，通过恰当遮挡原图的一部分，网络更倾向于深入学习图片当中的语义信息。于是cutout不再是纯粹抵抗遮挡的功能，而是起到数据增强的效果。

results

实验表明cutout有两个关键之处：

1. 必须仔细寻找和精确设定遮挡区域的大小。对于cifar10而言是16个pixel为边长的方形区域；对于cifar100而言则变为8个pixel。
2. 必须允许遮挡残缺的存在，也即允许遮挡区域的生成在图的外面。

注意以上tricks后得到如下的结果：

Method	C10	C10+	C100	C100+	SVHN
ResNet18 [5]	10.63 ± 0.26	4.72 ± 0.21	36.68 ± 0.57	22.46 ± 0.31	-
ResNet18 + cutout	9.31 ± 0.18	3.99 ± 0.13	34.98 ± 0.29	21.96 ± 0.24	-
WideResNet [22]	6.97 ± 0.22	3.87 ± 0.08	26.06 ± 0.22	18.8 ± 0.08	1.60 ± 0.05
WideResNet + cutout	$\mathbf{5.54 \pm 0.08}$	3.08 ± 0.16	$\mathbf{23.94 \pm 0.15}$	18.41 ± 0.27	$\mathbf{1.30 \pm 0.03}$
Shake-shake regularization [4]	-	2.86	-	15.85	-
Shake-shake regularization + cutout	-	$\mathbf{2.56 \pm 0.07}$	-	$\mathbf{15.20 \pm 0.21}$	-

Table 1: Test error rates (%) on CIFAR (C10, C100) and SVHN datasets. "+" indicates standard data augmentation (mirror + crop). Results averaged over five runs, with the exception of shake-shake regularization which only had three runs each. Baseline shake-shake regularization results taken from [4].

可见shake-shake + cutout达到了2.56的cifar10 state-of-the-art，超过了上面的shakedrop。

经过综合的实验发现，在cifar上文章给出的参数确实有奇效，作用于很多backbone都能稳定涨点。也许cutout能成为小图片数据任务的关键增强技术之一。

2. Mixup

paperlink

- Zhang H, Cisse M, Dauphin Y N, et al. mixup: Beyond empirical risk minimization[J]. arXiv preprint arXiv:1710.09412, 2017.

keys

本文描述了一种直观上十分魔幻的增强技术，同时对data和label进行线性组合后，作为新的增强后数据输入网络训练。

Contribution Motivated by these issues, we introduce a simple and data-agnostic data augmentation routine, termed *mixup* (Section 2). In a nutshell, *mixup* constructs virtual training examples

$$\tilde{x} = \lambda x_i + (1-\lambda)x_j, \quad \text{where } x_i, x_j \text{ are raw input vectors}$$
$$\tilde{y} = \lambda y_i + (1-\lambda)y_j, \quad \text{where } y_i, y_j \text{ are one-hot label encodings}$$

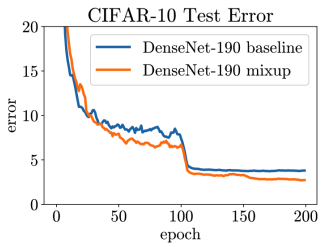
(x_i, y_i) and (x_j, y_j) are two examples drawn at random from our training data, and $\lambda \in [0, 1]$. Therefore, *mixup* extends the training distribution by incorporating the prior knowledge that linear interpolations of feature vectors should lead to linear interpolations of the associated targets. *mixup* can be implemented in a few lines of code, and introduces minimal computation overhead.

results

可以看到在cifar上，mix up能够实现明显的涨点。

Dataset	Model	ERM	*mixup*
CIFAR-10	PreAct ResNet-18	5.6	**4.2**
	WideResNet-28-10	3.8	**2.7**
	DenseNet-BC-190	3.7	**2.7**
CIFAR-100	PreAct ResNet-18	25.6	**21.1**
	WideResNet-28-10	19.4	**17.5**
	DenseNet-BC-190	19.0	**16.8**

(a) Test errors for the CIFAR experiments.



(b) Test error evolution for the best ERM and *mixup* models.

Figure 3: Test errors for ERM and *mixup* on the CIFAR experiments.

另外，从文中的实验还能看到，mix up能够赋予网络更强的对抗label corruption和adversarial attack的能力，且能让GANs的训练更佳稳定。

3. Auto Augment

paperlink

- Cubuk E D, Zoph B, Mane D, et al. AutoAugment: Learning Augmentation Policies from Data[J]. arXiv preprint arXiv:1805.09501, 2018.

keys

AA是非常强大的工具。

我们之前常见的数据增强都遵循一定的套路，这些套路是经典文章中使用的方式，具有较好的效果。例如imagenet中的RandomResizedCrop + RandomHorizontalFlip等。但是并没有人证明这就是最好的策略。

本文的作者提出利用RNN来对不同的数据集进行数据增强的搜索，以获得针对性的最佳策略。（Imagenet上的一种好策略如下）

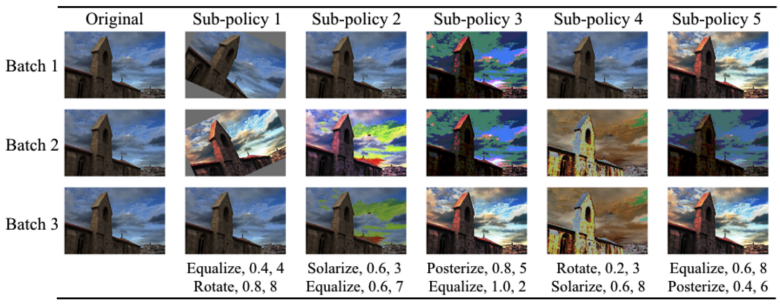


Figure 2: One of the successful policies on ImageNet. As described in the text, most of the policies found on ImageNet used color-based transformations.

实际上，上图中的策略是从下设的搜索空间里，利用RL搜索获得的。

Operation Name	Description	Range of magnitudes
ShearX(Y)	Shear the image along the horizontal (vertical) axis with rate *magnitude*.	[-0.3,0.3]
TranslateX(Y)	Translate the image in the horizontal (vertical) direction by *magnitude* number of pixels.	[-150,150]
Rotate	Rotate the image *magnitude* degrees.	[-30,30]
AutoContrast	Maximize the the image contrast, by making the darkest pixel black and lightest pixel white.	
Invert	Invert the pixels of the image.	
Equalize	Equalize the image histogram.	
Solarize	Invert all pixels above a threshold value of *magnitude*.	[0,256]
Posterize	Reduce the number of bits for each pixel to *magnitude* bits.	[4,8]
Contrast	Control the contrast of the image. A *magnitude*=0 gives a gray image, whereas *magnitude*=1 gives the original image.	[0.1,1.9]
Color	Adjust the color balance of the image, in a manner similar to the controls on a colour TV set. A *magnitude*=0 gives a black & white image, whereas *magnitude*=1 gives the original image.	[0.1,1.9]
Brightness	Adjust the brightness of the image. A *magnitude*=0 gives a black image, whereas *magnitude*=1 gives the original image.	[0.1,1.9]
Sharpness	Adjust the sharpness of the image. A *magnitude*=0 gives a blurred image, whereas *magnitude*=1 gives the original image.	[0.1,1.9]
Cutout [25, 73]	Set a random square patch of side-length *magnitude* pixels to gray.	[0,60]
Sample Pairing [51, 74]	Linearly add the image with another image (selected at random from the same mini-batch) with weight *magnitude*, without changing the label.	[0, 0.4]

Table 7: List of all image transformations that the controller could choose from during the search. Additionally, the values of magnitude that can be predicted by the controller during the search for each operation at shown in the third column (for image size 331x331). Some transformations do not use the magnitude information (e.g. Invert and Equalize).

通过RL（分组和搜索的细节不表），我们最终能对每个数据集获得25组增强策略；每组策略分为两个Operation，增强的时候依次进行即可。

例如在cifar上搜索获得的一组最佳策略如下：

	Operation 1	Operation 2
Sub-policy 0	(Invert,0.1,7)	(Contrast,0.2,6)
Sub-policy 1	(Rotate,0.7,2)	(TranslateX,0.3,9)
Sub-policy 2	(Sharpness,0.8,1)	(Sharpness,0.9,3)
Sub-policy 3	(ShearY,0.5,8)	(TranslateY,0.7,9)
Sub-policy 4	(AutoContrast,0.5,8)	(Equalize,0.9,2)
Sub-policy 5	(ShearY,0.2,7)	(Posterize,0.3,7)
Sub-policy 6	(Color,0.4,3)	(Brightness,0.6,7)
Sub-policy 7	(Sharpness,0.3,9)	(Brightness,0.7,9)
Sub-policy 8	(Equalize,0.6,5)	(Equalize,0.5,1)
Sub-policy 9	(Contrast,0.6,7)	(Sharpness,0.6,5)
Sub-policy 10	(Color,0.7,7)	(TranslateX,0.5,8)
Sub-policy 11	(Equalize,0.3,7)	(AutoContrast,0.4,8)
Sub-policy 12	(TranslateY,0.4,3)	(Sharpness,0.2,6)
Sub-policy 13	(Brightness,0.9,6)	(Color,0.2,8)
Sub-policy 14	(Solarize,0.5,2)	(Invert,0.0,3)
Sub-policy 15	(Equalize,0.2,0)	(AutoContrast,0.6,0)
Sub-policy 16	(Equalize,0.2,8)	(Equalize,0.6,4)
Sub-policy 17	(Color,0.9,9)	(Equalize,0.6,6)
Sub-policy 18	(AutoContrast,0.8,4)	(Solarize,0.2,8)
Sub-policy 19	(Brightness,0.1,3)	(Color,0.7,0)
Sub-policy 20	(Solarize,0.4,5)	(AutoContrast,0.9,3)
Sub-policy 21	(TranslateY,0.9,9)	(TranslateY,0.7,9)
Sub-policy 22	(AutoContrast,0.9,2)	(Solarize,0.8,3)
Sub-policy 23	(Equalize,0.8,8)	(Invert,0.1,3)
Sub-policy 24	(TranslateY,0.7,9)	(AutoContrast,0.9,1)

Table 8: AutoAugment policy found on reduced CIFAR-10.

注意，在每次训练的时候，对每个batch的数据，都要随机抽取一组策略进行增强操作。

results

在cifar10上的结果如下，这次是和pyramid-shakedrop结合。如果复现成果的话，基本已经把cifar10刷爆了。

Model	Baseline	Cutout [25]	AutoAugment
Wide-ResNet-28-10 [57]	3.87	3.08	2.68
Shake-Shake (26 2x32d) [59]	3.55	3.02	2.47
Shake-Shake (26 2x96d) [59]	2.86	2.56	1.99
Shake-Shake (26 2x112d) [59]	2.82	2.57	1.89
AmoebaNet-B (6,128) [21]	2.98	2.13	1.75
PyramidNet+ShakeDrop [60]	2.67	2.31	**1.48**

Table 1: Test set error rates (%) on CIFAR-10. Lower is better. All the results of the baseline models, and baseline models with Cutout are replicated in our experiments and match the previously reported results [25, 57, 59, 60]. One exception is Shake-Shake (26 2x112d), which has more filters than the biggest model in [59] – 112 vs 96, and the results were not previously reported. Note that the best policy is found on reduced CIFAR-10. See text for more details.

在imagenet上的结果如下，涨点也非常明显。

Model	Baseline	Inception Pre-processing [14]	AutoAugment
ResNet-50 [15]	24.70 / 7.80	23.69 / 6.92	22.37 / 6.18
ResNet-200 [15]	-	21.52 / 5.85	20.00 / 4.99
AmoebaNet-B (6,190) [21]	-	17.80 / 3.97	17.25 / 3.78
AmoebaNet-C (6,228) [21]	-	16.90 / 3.90	**16.46 / 3.52**

Table 5: Validation set Top-1 / Top-5 error rates (%) on ImageNet. Lower is better. ResNet-50 with baseline augmentation result is taken from [15]. AmoebaNet-B,C results with Inception-style preprocessing are replicated in our experiments and match the previously reported result by [21]. There exists a better result of 14.6% Top-1 error rate [68] but their method makes use of a large amount of weakly labeled extra data.

Insights

实际上，网络正则化和数据增强对训练结果的影响堪比backbone的结构设计。现在综合来看存在两个值得思考的问题：

1. 数据增强和正则化是否存在相似的数学本质？能否互相借鉴和融合？例如shake-drop中提出了下表，可以说是一个不错的开始

Table 1. Regularization methods generating new data. “Sample-wise generation” means that data is generated using single sample.

Regularization method	Data augmentation			Sample-wise generation
	In (input) data space	In feature space	In label space	
Data augmentation [16]	✓			✓
Adversarial training [6]	✓			✓
Label smoothing [23]			✓	✓
Mixup [31, 24]	✓		✓	
Manifold mixup [25]	✓	✓	✓	
Shake-Shake [5, 4]		✓		✓
ShakeDrop		✓		✓

2. 能否用一种方式，给出现有如此多技术的“完美”融合，或者针对不同给出一个“最佳组合”？