NIPS 2018 | Quoc Le 提出卷积网络专属正则化方法 DropBlock

机器之心编译
作者：Golnaz Ghiasi、Tsung-Yi Lin、Quoc V. Le
参与：路

近日，谷歌大脑团队在 arXiv 上发布论文，提出了一种卷积网络正则化方法 DropBlock，它是 dropout 的变体，但青出于蓝而胜于蓝。

深度神经网络在具备大量参数、使用大量正则化和噪声时效果很好，如权重衰减和 dropout。尽管 dropout 的首次成功与卷积网络相关，但近期的卷积架构很少使用 dropout。大部分情况下，dropout 主要用于卷积网络的全连接层。

本论文认为 dropout 的主要缺陷在于它随机丢弃特征。尽管这对全连接层有效，但对特征具备空间关联的卷积层而言没那么有效。当特征互相关联时，即使使用 dropout，输入信息仍然能传输到下一层，导致网络过拟合。这表明我们需要 dropout 的更结构化形式来更好地正则化卷积网络。

本论文介绍了一种 dropout 的结构化形式 DropBlock，对于正则化卷积网络格外有效。在 DropBlock 中，同一模块中的特征会被一起丢弃，即特征图的相邻区域也被丢弃了。由于 DropBlock 丢弃了相关区域中的特征，该网络必须从其他地方寻找证据来拟合数据（见图 1）。

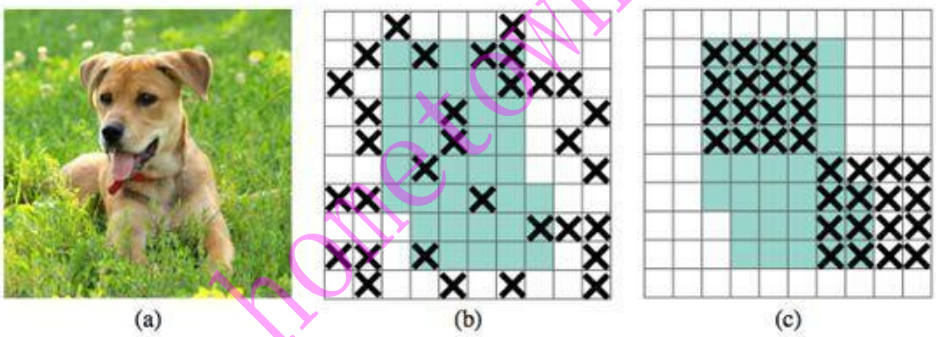

图 1 :(a) 卷积神经网络的输入图像。(b) 和 (c) 中的绿色区域包括激活单元，其包含输入图像中的语义信息。随机丢去激活单元在移除语义信息方面并没有效果，因为附近的激活单元包含高度相关的信息。而丢弃相邻区域可以移除特定语义信息（如头或脚），使剩余的单元学习可以分类输入图像的特征。dropblock 的思想是随机找一些点，然后自定义一个区域（block）把这里的信息一股脑全扔了。这样语义信息就不会冗余，从一定程度上使学习到的特征更加鲁棒。

实验中，DropBlock 在大量模型和数据集中的性能大大优于 dropout。向 ResNet-50 架构添加 DropBlock 使其在 ImageNet 数据集上的图像分类准确率从 76.51% 提升到 78.13%。在 COCO 检测任务上，DropBlock 将 RetinaNet 的 AP 从 36.8% 提升到 38.4%。

DropBlock: A regularization method for convolutional networks

Golnaz Ghiasi
Google Brain

Tsung-Yi Lin
Google Brain

Quoc V. Le
Google Brain

DropBlock: A regularization method for convolutional networks

摘要：深度神经网络在过参数化和使用大量噪声和正则化（如权重衰减和 dropout）进行训练时往往性能很好。dropout 广泛用于全连接层的正则化，但它对卷积层的效果没那么好。原因可能在于卷积层中的激活单元是空间关联的，使用 dropout 后信息仍然能够通过卷积网络传输。因此我们需要 dropout 的一种结构化变体来对卷积网络进行正则化。本论文就介绍了这样一种变体 DropBlock，它会丢弃特征图相邻区域中的单元。此外，在训练过程中逐渐增加丢弃单元的数量会带来更高的准确率，使模型对超参数选择具备更强的鲁棒性。大量实验证明，DropBlock 在正则化卷积网络方面性能优于 dropout。在 ImageNet 分类任务上，具备 DropBlock 的 ResNet-50 架构达到了 78.13% 的准确率，比基线模型提高了 1.6%。在 COCO 检测任务上，DropBlock 将 RetinaNet 的 AP 从 36.8% 提升到 38.4%。

DropBlock 是类似 dropout 的简单方法。二者的主要区别在于 DropBlock 丢弃层特征图的相邻区域，而不是丢弃单独的随机单元。Algorithm 1 展示了 DropBlock 的伪代码。

DropBlock 具备两个主要参数 block_size 和 γ。block_size 是要丢弃的 block 的大小（即该 block 置为 0），γ 控制要丢弃的激活单元的数量（注意这里的神经元不是真正丢了，而是某一次不用它的概率）。

我们在不同特征通道上对共享 DropBlock mask 进行了实验，也在每个特征通道上对 DropBlock mask 进行了实验。Algorithm 1 对应后者，它的效果在实验中也更好一些。

Algorithm 1 DropBlock

Input: output activations of a layer (A), $block\_size$, γ, $mode$
if $mode == Inference$ **then**
  return A
end if
Randomly sample mask M: $M_{i,j} \sim Bernoulli(\gamma)$
For each zero position $M_{i,j}$, create a spatial square mask with the center being $M_{i,j}$, the width, height being $block\_size$ and set all the values of M in the square to be zero (see Figure 2).
Apply the mask: $A = A \times M$
Normalize the features: $A = A \times \textbf{count}(M)/\textbf{count\_ones}(M)$

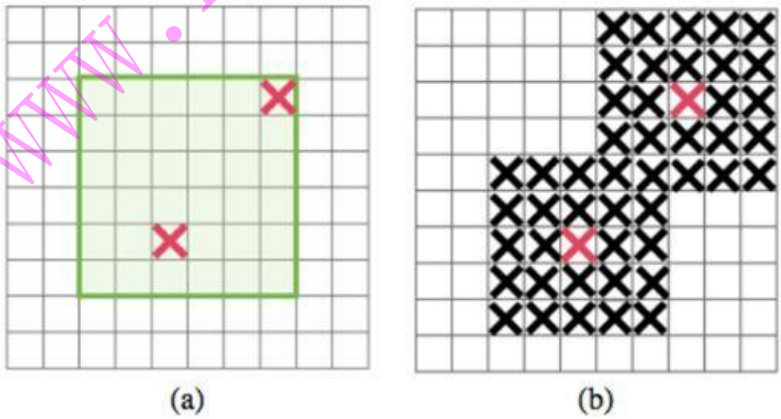

图 2：DropBlock 中的掩码采样（mask sampling）。(a) 与 dropout 类似，我们先在每个特征图上采样掩码 M。我们仅从绿色框中采样掩码，该区域中的每个采样条目（sampled entry）都可以扩展到完全包含在特征图中的掩码。(b) M 中每个 zero entry 都可以扩展为 block_size × block_size zero block。Blocksize 设置为 1 的时候和 dropout 类似，但是只在图中绿色区域丢

设置 block_size 的值。在实现中，我们为所有特征图设置常数 block_size，无论特征图的分辨率是多少。当 block_size = 1 时，DropBlock 类似 dropout，当 block_size 覆盖完整特征图的时候，DropBlock 类似 SpatialDropout。

设置 γ 的值。在实践中，我们没有显性地设置 γ 的值。如前所述，γ 控制要丢弃的特征的数量。假设我们想把每个激活单元的保留概率设置为 keep_prob，则在 dropout 中二进制掩码会被使用伯努利分布进行采样。但是，由于掩码中的每个 zero entry 将使用 block_size^2 进行扩展，得到的 block 将被完全包含在特征图中，因此我们在采样初始二进制掩码时需要据此调整 γ 的值。在我们的实现中，γ 可以按照下列公式计算：

$$\gamma = \frac{1 - keep\_prob}{block\_size^2} \frac{feat\_size^2}{(feat\_size - block\_size + 1)^2}$$

其中 keep_prob 是传统 dropout 中单元被保留的概率。有效种子区域的大小是 (feat_size-block_size+1)^2，其中 feat_size 是特征图的大小。dropblock 的奥妙在于被丢弃的 block 会有一些重叠，因此上述公式只是近似。实验中，我们首先估计 keep_prob 的值（0.75 到 0.95 之间），容纳后根据上述公式计算 γ 的值。

Scheduled DropBlock。我们发现具备固定 keep_prob 的 DropBlock 在训练过程中表现不好。最初 keep_prob 的值过小会影响模型的学习。而逐渐降低 keep_prob 的值（从 1 下降到目标值）更具鲁棒性，改进了大多数 keep_prob 的值。实验中，我们使用线性机制来降低 keep_prob 的值，其在很多超参数设置中都表现良好。该线性机制类似于 ScheduledDropPath。

表 2：AmoebaNet-B 架构在 ImageNet 数据集上的 top-1 和 top-5 验证准确率。

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

表 1：ResNet-50 架构在 ImageNet 数据集上的验证准确率。对于 dropout、DropPath 和 SpatialDropout，我们使用不同的 keep_prob 值进行训练，报告的是最优结果。DropBlock 使用 block_size = 7 进行训练。上表显示的是 3 次运行的平均值。

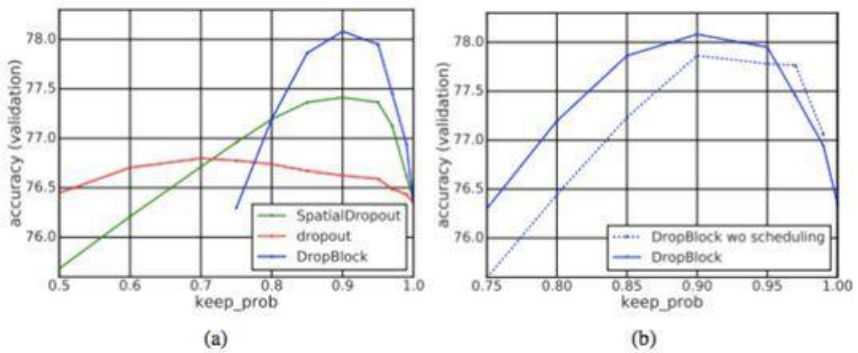

图 3：x 轴为 keep_prob，ResNet-50 模型在 ImageNet 数据集上的验证准确率变化（y 轴）。所有方法丢弃的都是第 3、4 组的激活单元。

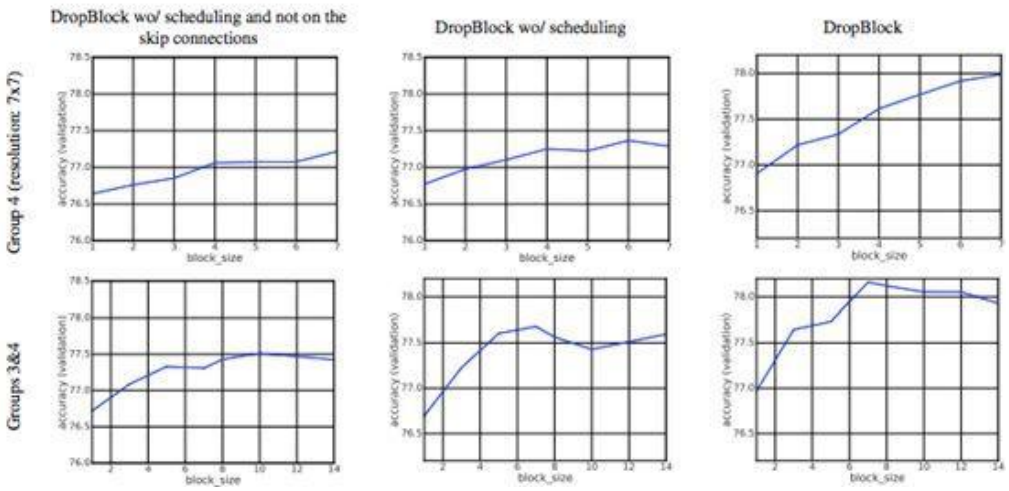

图 4 :在 ImageNet 数据集上训练的 ResNet-50 对比，DropBlock 应用于 group 4 或 groups 3、4。

总结

论文用翔实的实验来证明了 dropblock 的有效性。经过实验证明，block_size 控制为 7*7 效果最好，对于所有的 featuremap 都一样，γ \gammaγ 通过一个公式来控制，keep_prob 则是一个线性衰减过程，从最初的 1 到设定的阈值，论文通过实验表明这种方法效果最好。

Installation

Install directly from PyPI(python3):

```
pip install dropblock
```

or the bleeding edge version from github:

```
pip install git+https://github.com/miguelvr/dropblock.git#egg=dropblock
```

local install(require numpy torch-0.4.1):

```
python setup.py install
```

参考链接：

1. http://www.sohu.com/a/274221901_129720 （原文）
2. https://arxiv.org/pdf/1810.12890.pdf （paper）
3. http://www.cnblogs.com/xiaohuahua108/p/9977295.html
4. https://swift.ctolib.com/miguelvr-dropblock.html (软件安装)