神经网络压缩-剪枝方法的分析与探究

主要内容

- 模型压缩的方法介绍
- 5种经典剪枝方法的分析
- 目前我的研究进展

模型压缩的几种方法介绍。

- 1.参数共享。主要思想就是让几个参数共享一个值。（比如哈希网络等）
- 2.量化。量化主要思想是降低权重所需要的比特数来压缩原始网络
  3.网络分解。主要思想就是将一个大的二维矩阵，分解成三维卷积核或者使用多个一维张量外积求和。
- 4.轻量化网络设计。MobileNet，ShuffleNet 等
- 我们涉及的方法。
- 1.蒸馏
- 2.剪枝

知识蒸馏

知识蒸馏就是用教师网络的输出信息，指导学生网络参数的训练。

- Hinton 等引入知识蒸馏压缩框架，使用教师模型最终 softmax 的输出对学生模型的输出进行监督，使得教师模型的信息可以传递到学生模型中。知识蒸馏算法虽然简单，但在各种图像分类任务中显示出良好效果。
- Romero 等提出当模型层数较多时，直接使用教师模型的输出对学生模型进行监督会比较困难，因此提出 Fitnets 模型，将教师模型的中间层输出作为对学生模型的中级监督信息。马师兄的论文，就是用到了中间层学习。
- 我们做的kjw的项目，训练方法上就用了hinton的知识蒸馏训练网络。

知识蒸馏[hinton]：

学习方法：$L_{kd}(t,s) = (1-\alpha)L_{CE}\left(y,\sigma(s)\right) + 2\mathrm{T}^2 L_{kl}(\sigma\left(\frac{t}{T}\right),\sigma\left(\frac{s}{T}\right)$

类别	牛	狗	猫	汽车
硬目标	0	1	0	0
Softmax 输出	10^{-6}	0.9	0.1	10^{-9}
软化目标	0.05	0.3	0.2	0.005

Hinton14年的文章使用软化目标体现不同类别之间的相似性，增加了监督信息，因此准确度更好，如图狗与猫和牛的接近程度远大于狗与汽车，软化目标在在数值上也能够体现。

剪枝方法介绍

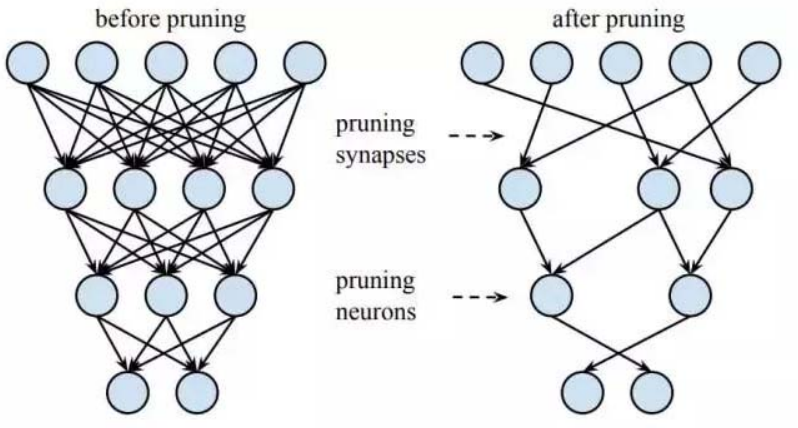
- 剪枝的主要思想：

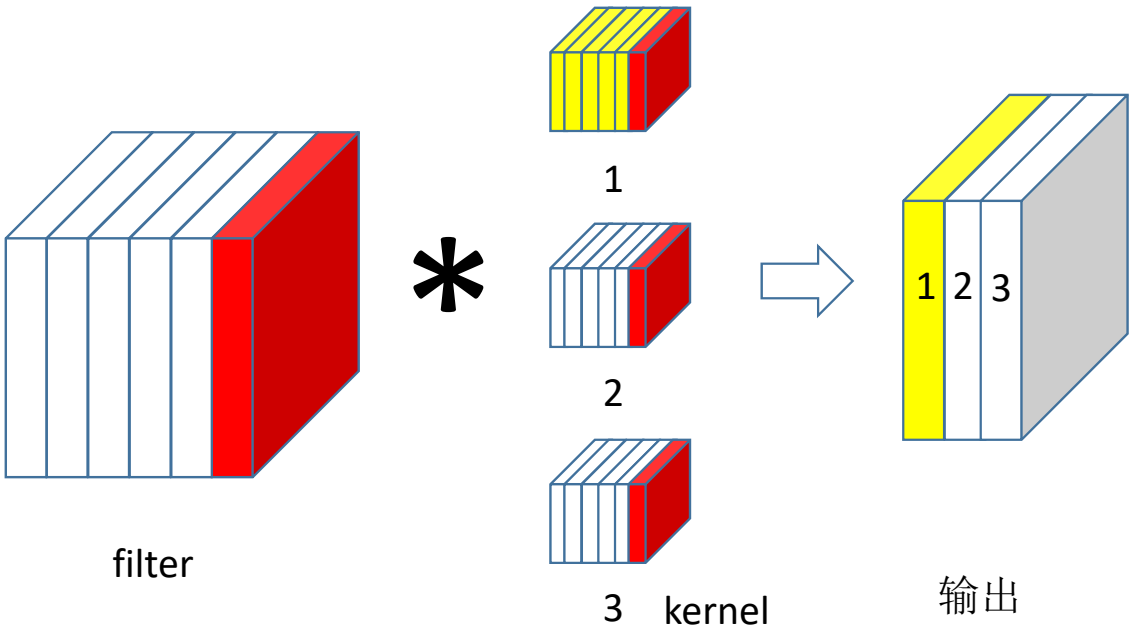
现代深度神经网络使用参数已足够强大的模型，这些模型的参数有一定冗余的参数，甚至有对模型表现力贡献度影响不大的参数。因此我们可以对模型进行剪枝。剪枝的本质就是，找到这些冗余参数，或者是对模型贡献度不大的参数进行删减。因此可以将剪枝归结于，对模型参数的贡献度做一个排序，删除不重要的，留下重要的参数。

- 剪枝分类

1.稀疏化剪枝（非结构化剪枝）

2.结构化剪枝（卷积核和filter剪枝。）





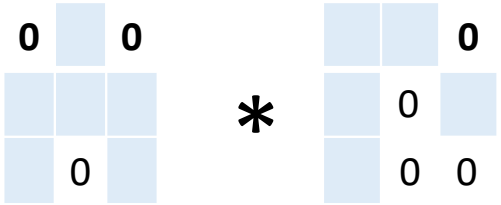
稀疏化剪枝

DEEP COMPRESSION: COMPRESSING DEEP NEURAL NETWORKS WITH PRUNING, TRAINED QUANTIZATION AND HUFFMAN CODING 2016ICLR 很多奖

- 在初始化训练阶段后，我们通过移除权重低于阈值的连接而实现 DNN 模型的剪枝，这种剪枝将密集层转化为稀疏层。
- 稀疏化剪枝的优劣性：

局限性：稀疏化剪枝能够大大压缩结构的参数量。但是DNN是卷积操作。虽然稀疏了矩阵，但是矩阵的计算量在没有通过三方的加速方法下，并没有直接实质性的对DNN的操作进行加速。

优势：由于在剪枝过程中，没有相关结构性的限制，剪枝可以大大剪掉参数量，韩松的论文中能够实现指标35-49倍，而模型的表现型没有损失



图中蓝色的方块，表示阀值以上的权重，0表示阀值以下的权重，应当删除，后来论文实验表明，这种训练过程除了可以学习神经网络的权重外，还可以学习神经元间的连通性。（既神经网络之间的结构关系。）

结构化剪枝

- 为了能够对神经网络能够直接加速，目前文章中大部分的剪枝方式都采用结构化的剪枝方式。结构化剪枝由于在剪枝过程中有结构之间的限制，所以结构化剪枝剪枝的比例，没有非结构化剪枝的比例大。目前最好的能做到，剪掉resnet50/101百分之40精度损失1个点多一点。

经典论文分析：

PRUNING FILTERS FOR EFFICIENT CONVNETS——2017 ICLR

指标：在CIFAR10上，VGG-16的推理成本可降低高达34%，ResNet-110最高可降低38%，同时通过重新训练网络恢复接近原始精度。

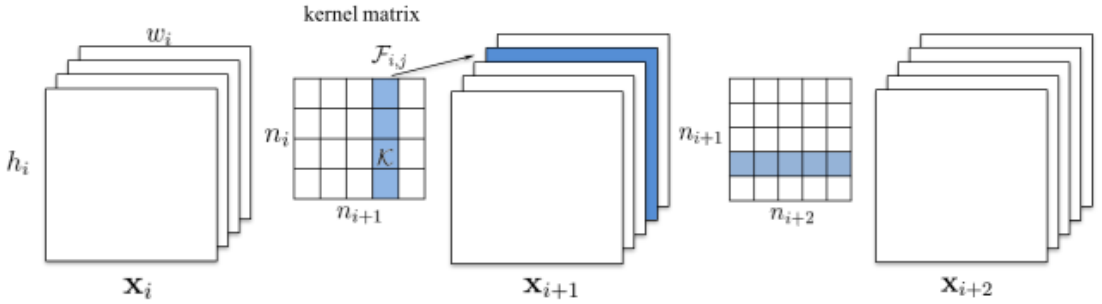
这篇文章提出了使用卷积核L1正则化的思路来判断卷积核的重要性。后续发展出了L0,L2正则化的剪枝方法。

这篇文章剪枝步骤如下：

1.对每个卷积核F，计算他的权重绝对值之和s；

2.根据s的大小在每一层卷积中进行排序

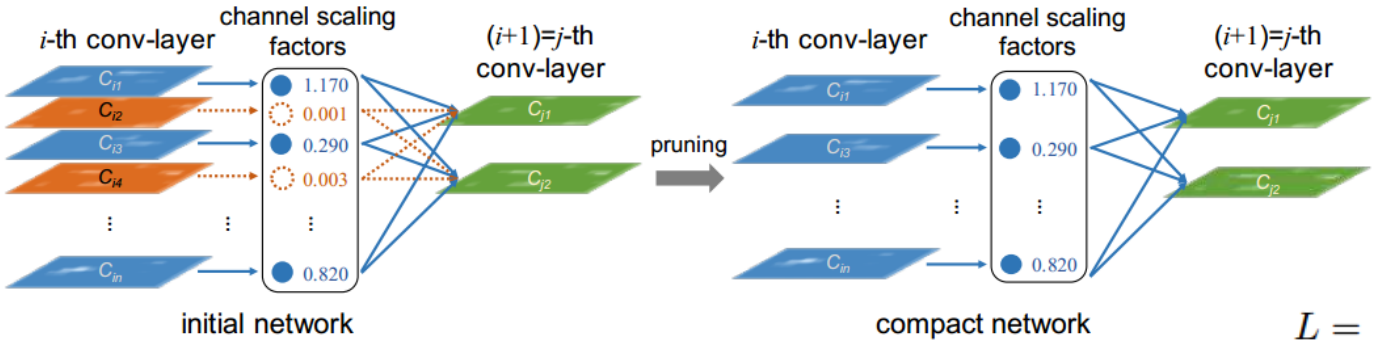
3.将m个卷积和s最小的卷积核以及对应的feature maps 剪掉。下一个卷积层中与剪掉的feature maps相对应的核也要移除；



Learning Efficient Convolutional Networks through Network Slimming

2017 ICCV

- batch normalization(BN)——在每次SGD时，通过mini-batch来对相应的activation做规范化操作，使得结果（输出信号各个维度）的均值为0，方差为1.而在这个规范操作过程中，就会对每一层的channel产生一个缩放因子γ
- Network slimming，利用BN层中的缩放因子γ，在训练过程当中来衡量channel的重要性，将不重要的channel进行删减，达到压缩模型大小，提升运算速度的效果。 看一下模型图，左边为训练当中的模型，中间一列是scaling factors，也就是BN层当中的缩放因子γ，当γ较小时（如图中0.001,0.003），所对应的channel就会被删减，得到右边所示的模型。 道理非常简单，而且巧妙的将γ增加到目标函数中去，达到了一边训练一边剪枝的奇效。



$$L = \sum_{(x,y)} l(f(x, W), y) + \lambda \sum_{\gamma \in \Gamma} g(\gamma)$$

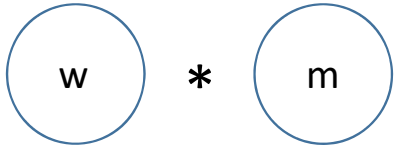
这篇文章，还有一个比较创新的点，将缩放因子加入到loss中，
使得在训练过程中，loss减少的同时缩放因子也跟着减少，这个
创新，后来的很多文章都借鉴了这个思想，将权重或者其他的加入到loss中，
使得训练过程中权重w更加稀疏话，能够更好的剪枝。

Soft Filter Pruning for Accelerating Deep Convolutional Neural Networks

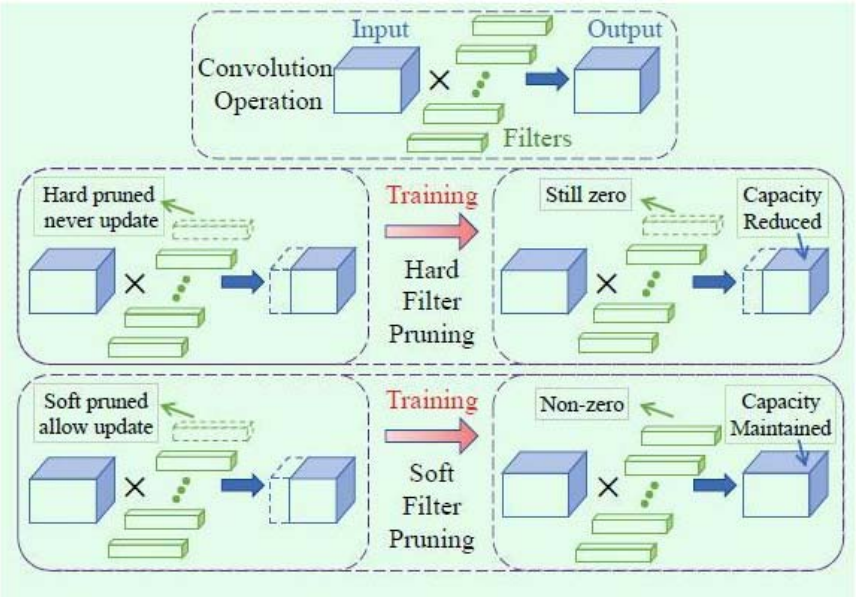
IJCAI 2018的关于模型加速压缩的文章

这篇文章思想简洁而且有效，主要是和Hard Filter Pruning（HFP）做对比，HFP是比较常见的剪枝方式，一般是按照某些指标对卷积核进行排序，然后直接剪掉不符合指标的卷积核，然后做fine tune，fine tune的时候网络中就不包含那些被剪掉的卷积核。这篇文章主要提出了通过Soft Filter Pruning (SFP)做模型加速，SFP和HFP的不同点在于剪掉的卷积核依然参与下一次迭代更新，而并不是剪掉一次就永远没有了，而且SFP在不采用fine tune的情况下依然能够有不错的效果，因为SFP在每个epoch结束后会进行剪枝，剪枝后就会再训一个epoch，然后继续剪枝，

1.这篇文章卷积核的判断他是采用L2，正则化的方法。
2.文章在训练过程中，使用了加mask的方法。



图中表示，网络中每一个权重w乘m（0,1）
如果改卷积核通过L2的正则化判断以后是需要删除的卷积核，那么就将m设置为0，表示该卷积核删除。
如果该卷积核需要保留，就设置为1.

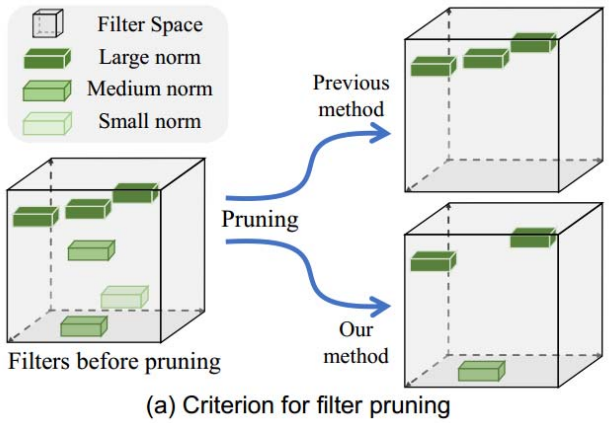


这种剪枝方法，在训练过程既是剪枝过程，不依赖预训练过程，关键指标如下：

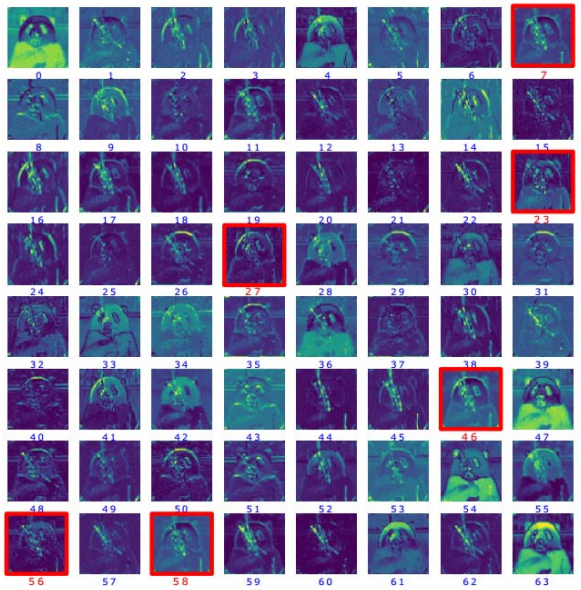
网络	drop（30%）
res18	3.18
res50	0.84
res101	-1.4

Filter Pruning via Geometric Median for Deep Convolutional Neural Networks Acceleration 2019 cvpr

- 文章指出，之前的剪枝评价指标，都是通过范数来判断卷积核的重要性，这实际上包含两个隐含条件1）范数标准差足够大；2）最小的范数接近于0。
- 文章提出了一种基于几何中心滤波器评价指标（FPGM）



(a) Criterion for filter pruning



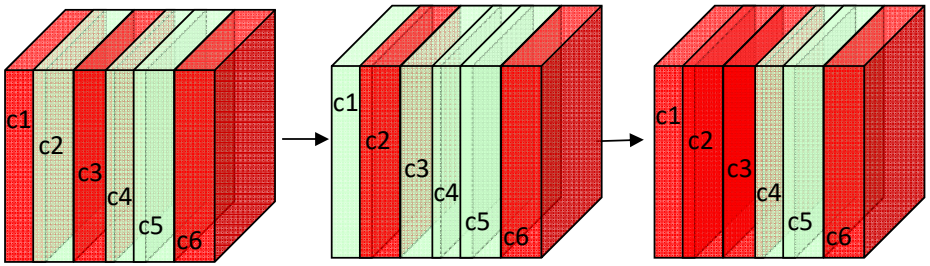
文章通过判断filter之间的欧几里得距离，
创造性删除距离较近的filter（既是图中的中位数filter）
文章对为什么要这样删除filter做出来解释：
对每一层的feature maps做出可视化，我们可以发现，
欧式距离比较近的两个filter之间是比较相似的。我们可以认为，相似的两个filter之间的信息会有冗余。

剪枝目前state-of-theart

Depth	Method	Fine-tune?	Baseline top-1 acc.(%)	Accelerated top-1 acc.(%)	Baseline top-5 acc.(%)	Accelerated top-5 acc.(%)	Top-1 acc. ↓(%)	Top-5 acc. ↓(%)	FLOPs↓(%)
18	MIL [5]	✗	69.98	66.33	89.24	86.94	3.65	2.30	34.6
	SFP [15]	✗	**70.28**	67.10	**89.63**	87.78	3.18	1.85	**41.8**
	Ours (FPGM-only 30%)	✗	**70.28**	67.78	**89.63**	88.01	2.50	1.62	**41.8**
	Ours (FPGM-mix 30%)	✗	**70.28**	**67.81**	**89.63**	**88.11**	**2.47**	**1.52**	**41.8**
	Ours (FPGM-only 30%)	✓	**70.28**	68.34	**89.63**	**88.53**	1.94	**1.10**	**41.8**
	Ours (FPGM-mix 30%)	✓	**70.28**	**68.41**	**89.63**	88.48	**1.87**	1.15	**41.8**
34	SFP [15]	✗	**73.92**	71.83	**91.62**	90.33	2.09	1.29	**41.1**
	Ours (FPGM-only 30%)	✗	**73.92**	71.79	**91.62**	**90.70**	2.13	**0.92**	**41.1**
	Ours (FPGM-mix 30%)	✗	**73.92**	**72.11**	**91.62**	90.69	**1.81**	0.93	**41.1**
	PFEC [21]	✓	73.23	72.17	-	-	**1.06**	-	24.2
	Ours (FPGM-only 30%)	✓	**73.92**	72.54	**91.62**	**91.13**	1.38	**0.49**	**41.1**
	Ours (FPGM-mix 30%)	✓	**73.92**	**72.63**	**91.62**	91.08	1.29	0.54	**41.1**
50	SFP [15]	✗	**76.15**	74.61	**92.87**	92.06	1.54	0.81	41.8
	Ours (FPGM-only 30%)	✗	**76.15**	**75.03**	**92.87**	**92.40**	**1.12**	**0.47**	42.2
	Ours (FPGM-mix 30%)	✗	**76.15**	74.94	**92.87**	92.39	1.21	0.48	42.2
	Ours (FPGM-only 40%)	✗	**76.15**	74.13	**92.87**	91.94	2.02	0.93	**53.5**
	ThiNet [25]	✓	72.88	72.04	91.14	90.67	0.84	0.47	36.7
	SFP [15]	✓	**76.15**	62.14	**92.87**	84.60	14.01	8.27	41.8
	NISP [39]	✓	-	-	-	-	0.89	-	44.0
	CP [16]	✓	-	-	92.20	90.80	-	1.40	50.0
	Ours (FPGM-only 30%)	✓	**76.15**	**75.59**	**92.87**	**92.63**	**0.56**	0.24	42.2
	Ours (FPGM-mix 30%)	✓	**76.15**	75.50	**92.87**	92.63	0.65	**0.21**	42.2
	Ours (FPGM-only 40%)	✓	**76.15**	74.83	**92.87**	92.32	1.32	0.55	**53.5**
101	Rethinking [38]	✓	**77.37**	75.27	-	-	2.10	-	**47.0**
	Ours (FPGM-only 30%)	✓	**77.37**	**77.32**	**93.56**	**93.56**	**0.05**	**0.00**	42.2

我目前工作

- 层与层之间关系，与层间关系。



layer1　　layer2　　layer3

Layer1，layer2，layer3之间关系表示为层与层之间的关系（vgg16—16层，resnet50—50层）
C1，c2，c3，c4，c5，c6表示层间关系。既是每一层之间各个通道之间的关系。

层与层之间关系，与层内关系重要性证明：

	L	B（164）	s	G
正常剪枝	93.1	94.92	92.26	93.43
剪枝得到层间结构，层内随机	92.5	94.11	91.98	93.10
随机层间结构	87.6	89.9	68.4	75.8
剪枝得到层间用l1层内选择	92.8		92.04	93.10

图中L，B，S，G分别表示上面提到的4中结构化剪枝
通过对表中数据分析：
1.随机层内结构，和随机层间结构对比，层间结构比层内结构重要的多。
2.将剪枝分为层间剪枝，和层内剪枝两部分。似乎可以我们在剪枝过程中，优先找到层间结构，层内结构可以通过简单的方法提升。

接下来工作

在2019年ICLR的两篇论文，[Gilpin L H](best paper),[Liu Z]。分析了剪枝过程留下来权重信息的重要性分析，

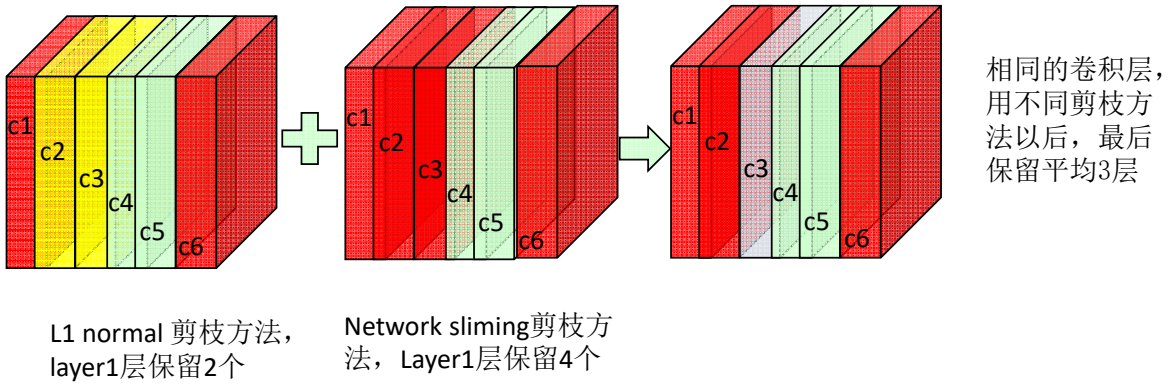
（剪枝过程，留下来的主要信息是网络之间结构信息，网络权重信息可以通过相关初始化获取）加上我上一页的ppt分析。我们似乎可以进一步得出一个结论：神经网络剪枝留下来的信息，主要是层与层之间的关系。

那么我可以将创新性的将剪枝方法，分为以下三步：

1.通过多种方法，剪枝相同比例。得到各自层与层之间的相互关系。

2.分析综合各种剪枝方法，得到每一层的剪枝比例。

3.然后用一个比较常见，简单的方法，选择层内之间的结构。



谢谢