# 网络压缩探幽（一）

陈超
南京大学

## 前言

这是我第一次涉猎网络压缩的方向，所以只能从经典的论文看起，稍晚才会接触最新的文章.
常用的网络压缩方法有：

- 奇异值分解 (SVD).
- 网络剪枝 (Network Pruning). 将小于一定阈值的权重置为 0，得到一个稀疏网络，再对这个网络训练以提高准确率.
- 结构设计. 用更小的网络达到相似的性能.
- 量化. 将网络中的浮点运算转化为定点运算，一般用 8 比特或更少的比特.
- ···

由于内容繁多，时间仓促，本周的目的是大致了解网络结构设计部分.

## 人工模块设计

### $1\times1$ 卷积

SqueezeNet[1] 是一个经典的轻量化网络，其设计思路如下：

- 用 $1\times1$ 卷积代替 $3\times3$ 卷积. 参数量变为原来的 $\frac{1}{9}$.
- 减少 $3\times3$ 卷积的输入通道.
- 延迟下采样. 这样有利于在参数量减少的情况下保持准确率，因为过早的下采样可能会丢失重要特征.

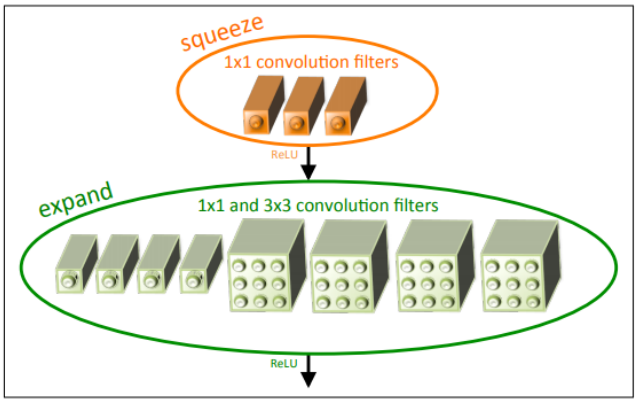


**Figure 1:** fire 模块. squeeze 部分用 3 个 1x1 的卷积核，expend 部分用 4 个 1x1 的卷积核和 4 个 3x3 的卷积和. 激活函数用 Relu.

SqueezeNet 由 fire 模块堆叠而成，如图 1所示，其精华部分是用 $1\times1$ 卷积减少通道数，看起来好像把通道 " 挤压 " 了一样. 同时借鉴了 GoogleNet 的 Inception 模块，作并行的两路卷积然后将结

果拼接 (concatenate) 起来.

### 分组卷积

AlexNet[2] 最早使用了分组卷积. 分组卷积把特征图分成 $G$ 组，分别用 $G$ 组卷积核进行卷积，再将结果拼接 (concatenate) 起来. 其参数量减少为原来的 $\frac{1}{G}$.

### 深度可分离卷积

深度可分离卷积最早由 MobileNet[3] 提出, 它包括 depthwise 和 pointwise 两个步骤，即按位相乘和 $1\times1$ 卷积. depthwise 等价于组数为 $M$ 的分组卷积. 设 $D_F$ 为特征图尺寸，$D_K$ 为卷积和尺寸，$M$ 为输入通道数，$N$ 为输出通道数，则普通卷积的计算量为 $D_F\times D_F\times D_K\times D_K\times M\times N$,. 而引入深度可分离卷积后，计算量减少为 $D_F\times D_F\times D_K\times D_K\times M+D_F\times D_F\times M\times N$.

### Channel Shuffle

为了解决分组卷积中组与组之间信息不交流的问题，ShuffleNet[4] 在分组卷积前先把特征图的通道打乱. 通道打乱不是随机地打乱，而是让新的任一组特征尽可能包含原先每组的特征. 具体操作为将通道 [C] 重塑 (reshape) 为 [G,c]，其中 G 为组数，c 为每组通道数,$C=G\times c$ 为总通道数. 接着转置 (transpose) 成 [c,G], 最后重塑为原来的 [C].

## 神经网络架构搜索

本部分内容主要参考 2018 年的一篇综述类文章 <Neural Architecture Search: A Survey>[5]. NAS(Neural Architecture Search) 是 Auto ML 的子任务，旨在自动搜索网络结构的较优超参数. 其抽象说明如图 2.

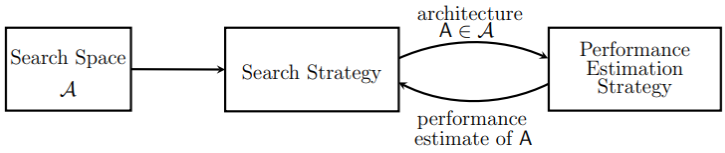


**Figure 2:** NAS 示意图. 搜索策略从搜索空间中选择一个网络结构交给评估模块，再接收评估结果，以确定网络结构是否最优和下一步的搜索方向.

网络压缩既要求高准确率又要求轻量，可以视为一个多目标的 NAS 问题.

### 搜索空间

搜索空间 (Search Space) 决定了哪些结构可以加到最后的网络中. 人为引入先验知识或设定好子结构能够有效的减小搜索空间, 简化搜索. 但同时可能引入偏好, 限制了网络学习到超越人类知识的结构.

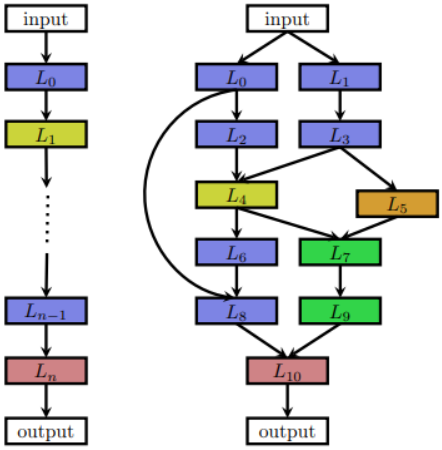


**Figure 3:** 左为链式结构，右为跳跃结构

一种简单的搜索空间为链式搜索空间，见图 3左，其参数为

- (最大) 层数 n.
- 层操作类型，e.g. 池化，卷积等.
- 层的超参数，如卷积层的核的数量尺寸、步长或者全连接层神经元的数量.

另一种搜索空间为含跳跃连接 (skip connection) 的搜索空间，见图 3右. ResNet 和 DenseNet 可视为该空间中的网络结构. 与链式搜索空间相比. 除了上述三个超参数，还多了描述层间是否有连接的超参数，搜索空间更为庞大.

现有的人工设计网络多由重复的单元组成，如 ResNet、GoogleNet、DenseNet，故有人提出搜索重复单元 (cells or blocks)，而不是搜索整个网络结构. 比如搜索两种 cell：normal cell 和 reduction cell. 其中 normal cell 不改变特征图尺寸而 reduction cell 会缩小特征图尺寸. 这种搜索空间有如下三个优点：

- 堆叠 cell 的方法在之前很多人工网络中被证明是行之有效的.
- 搜索空间极大减小.
- 易迁移.

一旦最优 cell 被搜索出来后，剩下的问题就是如何堆叠. 有的人采用线性结构并且每个 cell 接受前两个 cell 的输入 (个人理解有点像 GoogleNet 那样并行的组织方式)，有的人采用 DenseNet 那种密集跳跃连接的方式. 理论上，cell 之间可以被随意的连接.

## 搜索策略

一个良好的搜索策略 (Search Strategy) 应该尽可能快地找到性能较好的网络结构.

搜索策略很多，包括：随机搜索、贝叶斯优化、进化算法、强化学习和基于梯度的方法.

### 强化学习

基于强化学习的搜索策略可定义为: 搜索空间为 action space，从搜索空间选出一个网络结构为 action, 估计的网络性能为 reward. 常用的强化学习算法有 Policy Gradient、Q learning、Proximal Policy Optimization.

### 演化计算

演化算法过去曾用于搜索神经网络的参数. 当参数量越来越大时，随机梯度下降算法 (SGD) 优于演化算法. 现在演化算法仍然可用于网络结构搜索（不过根据历史的进程推断，仍然会被 SGD 替代).

总的思路是，初始化一组模型，在每个进化步骤中，采样至少一个模型并以一定概率突变．突变操作包括添加/删除层、改变层的超参数、跳过连接等．突变后对后代的性能进行评估，优胜劣汰．

细节处不同人的做法存在差异．例如，如何对父类进行采样、如何更新种群、如何产生后代等．(个人认为这里引入了太多的先验知识并且很难说哪种方法一定好于其他的．)

**梯度优化**

上述方法的搜索空间都是离散的．为了实现梯度优化，Liu 等[7] 提出连续松弛的搜索空间．设 $\sigma_i$ 表示某一操作 (比如卷积或池化)，x 为输入，给定一组操作集合 $\{\sigma_1, \cdots, \sigma_m\}$, 计算输出:

$$y = \sum_{i=1}^{m} \alpha_i \sigma_i(x), \alpha_i \geq 0, \sum_{i=1}^{m} \alpha_i = 1$$

其中 $\alpha_i$ 是待优化的系数．采用交替梯度下降的方法优化，即网络权重根据训练集的误差优化，$\alpha_i$ 根据验证集的误差优化，二者交替进行．最后重新离散化，取最优操作 $\sigma^* = \arg\max_{\sigma_i} \alpha_i$.

**评估策略**

一个简单的方法就是像之前一样对模型进行训练并验证其性能. 但是这很耗时. 现今很多研究关注于更简单的评估策略.

假设小规模训练的效果可以代表大规模训练的效果，则可以采取如下措施: 短时训练、在训练集上的子集训练、在低分辨率的图像上训练或者每层使用少的通道数．遗憾的是，有研究表明上述假设不总是成立.

另一种方法是基于学习曲线外推的方式．通过一开始的学习曲线预测最终的学习效果，提前终止曲线不好的模型的训练以达到加速的目的．有时候简单的外推算法预测效果不佳可以训练一个辅助的预测网络.

还有一种方法是用已训练好的权重初始化网络，这样就不用从头训练，能更快地收敛．网络态射 (network morphisms) 可以让小网络扩张为大网络的同时网络代表的函数不变，而近似网络态射 (approximate network morphisms) 可以使大网络收缩为小网络.

One-Shot Architecture Search 是第四种加速性能评估的方法，它将所有网络结构视为一个超图 (super graph) 的不同子图，并不同子图在超图中的共同边共享权重，见图 4．只有一个的超网络需要训练，然后通过从中继承权重来评估各个子网络的性能，这样便不再需要额外的训练．类似于上面提到的小规模训练，该方法也是用估计的网络性能去代表实际网络性能．目前没有研究证明这种估计是合理的.

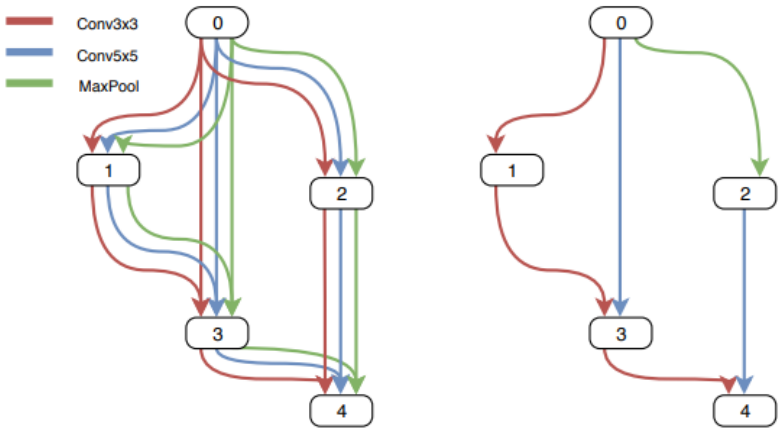


**Figure 4:** 以含 1 个输入节点 (0),3 个隐藏节点 (1,2,3),1 个输出节点 (4) 的网络为例. 不同颜色的线分别代表 3x3 卷积、5x5 卷积、最大池化. 左为超图，右为其中一个子图. 一旦超网络训练完成，子网络直接使用超网络的权重.

### 未来方向

NAS 有以下几个值得探索的方向：

- 更广泛的应用. 目前 NAS 研究主要针对图像分类. 在图像存储、语义分割、机器翻译等领域有一些应用. 用在 GAN 或感知融合是个不错的选择.
- 多任务、多目标问题.
- 与更多的强化学习方法结合. 强化学习是最热门、效果最好的搜索策略 (之一).
- 更通用、灵活的搜索空间.
- 统一的评价标准. 有些论文没有证明性能的提升究竟是源于训练技巧还是网络结构.
- 可解释性. 对搜索出来的最优网络结构，目前很难解释它为什么最优.
- ...

## 总结

前面提到的一些操作，除了 channel shuffle，剩下的 1x1 卷积、分组卷积、深度可分离卷积在各种神经网络中都很频繁的出现. 除非引入新的算子，再人为设计一种高效的操作感觉比较难，因为卷积翻来覆去就那几种玩法. NAS 应该是未来网络结构设计的主流，凡是关于深度学习的方向都可以引入 NAS. 欲复现 NAS 相关的论文加深理解，然而手头算力不够，只能日后再说. 未来一周的计划是了解网络压缩的另外几种方式，有条件的话复现一下其中的算法.

## References

[1] Iandola F N, Han S, Moskewicz M W, et al. SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and< 0.5 MB model size[J]. arXiv preprint arXiv:1602.07360, 2016.

[2] Krizhevsky A, Sutskever I, Hinton G E. Imagenet classification with deep convolutional neural networks[C]//Advances in neural information processing systems. 2012: 1097-1105.

[3] Howard A G, Zhu M, Chen B, et al. Mobilenets: Efficient convolutional neural networks for mobile vision applications[J]. arXiv preprint arXiv:1704.04861, 2017.

[4] Zhang X, Zhou X, Lin M, et al. Shufflenet: An extremely efficient convolutional neural network for mobile devices[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2018: 6848-6856.

[5] Elsken T, Metzen J H, Hutter F. Neural architecture search: A survey[J]. arXiv preprint arXiv:1808.05377, 2018.

[6] Cai H, Chen T, Zhang W, et al. Efficient architecture search by network transformation[C]//Thirty-Second AAAI conference on artificial intelligence. 2018.

[7] Liu H, Simonyan K, Yang Y. Darts: Differentiable architecture search[J]. arXiv preprint arXiv:1806.09055, 2018.