模型加速与压缩 | 剪枝乱炖

Colorjam
在浮动不安世界里找到安稳

剪枝是模型压缩的一个子领域，依据剪枝粒度可以分为非结构化/结构化剪枝，依据实现方法可以大致分为基于度量标准/基于重建误差/基于稀疏训练的剪枝，并且逐渐有向AutoML发展的趋势。由于实现方法在剪枝粒度上是有通用性的，本文主要从实现方法进行展开，康康近年来关于剪枝的有的没的，从个人角度对近几年经典的剪枝方法以及其拓展进行一下梳理。

基于度量标准的剪枝

这类方法通常是提出一个判断神经元是否重要的度量标准，依据这个标准计算出衡量神经元重要性的值，将不重要的神经元剪掉。在神经网络中可以用于度量的值主要分为3大块：**Weight / Activation / Gradient**。各种神奇的组合就产出了各种metric玩法。

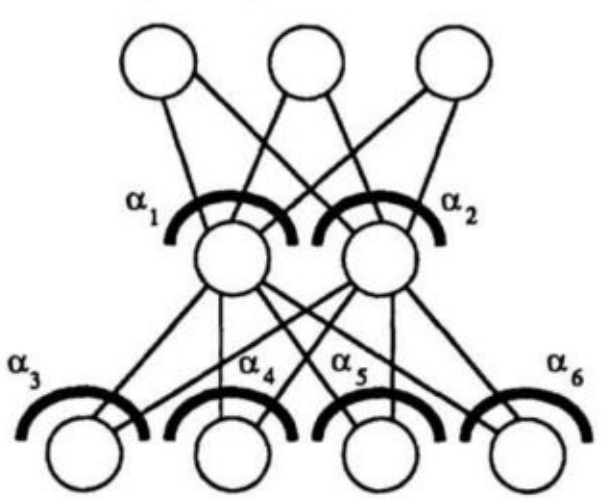
> 这里的神经元可以为非结构化剪枝中的单个weight亦或结构化剪枝中的整个filter。

Weight：基于结构化剪枝中比较经典的方法是Pruning Filters for Efficient ConvNets(ICLR2017)，基于L1-norm判断filter的重要性。Filter Pruning via Geometric Median for Deep Convolutional Neural Networks Acceleration(CVPR2019) 把绝对重要性拉到相对层面，认为与其他filters太相似的filter不重要。

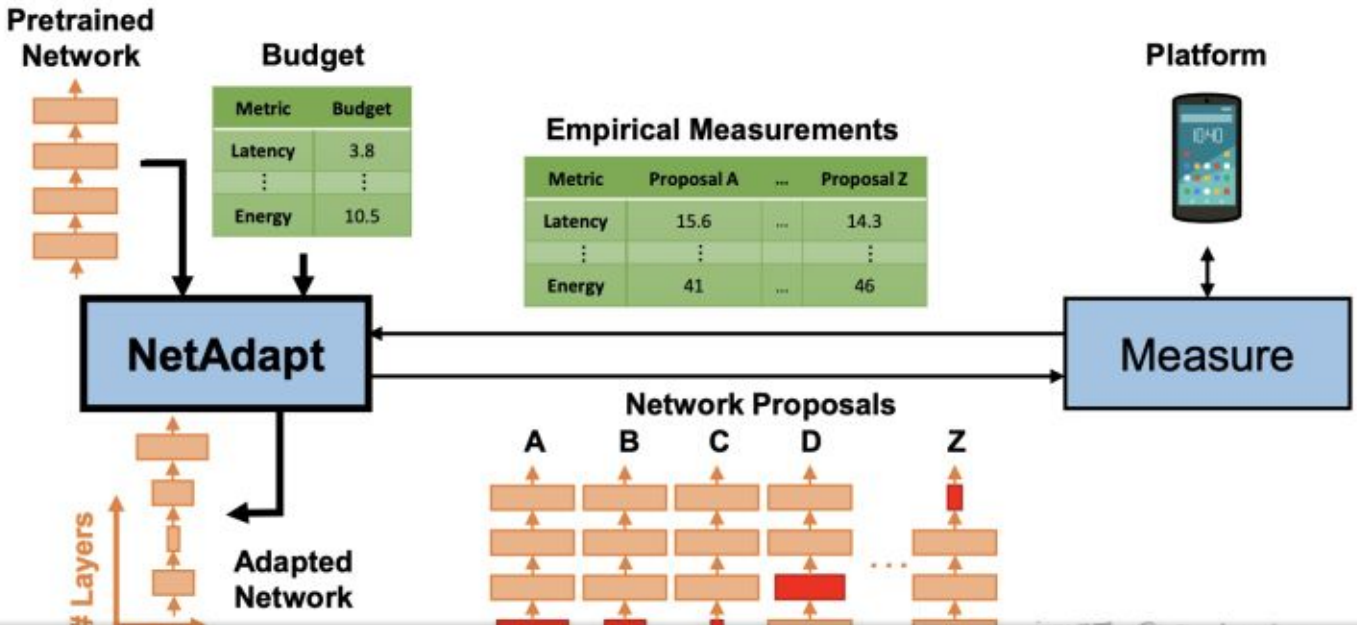
Activation：Network trimming: A data-driven neuron pruning approach towards efficient deep architectures 用activations中0的比例 (Average Percentage of Zeros, APoZ)作为度量标准，An Entropy-based Pruning Method for CNN Compression 则利用信息熵进行剪枝。

Gradient：这类方法通常从Loss出发寻找对损失影响最小的神经元。将目标函数用泰勒展开的方

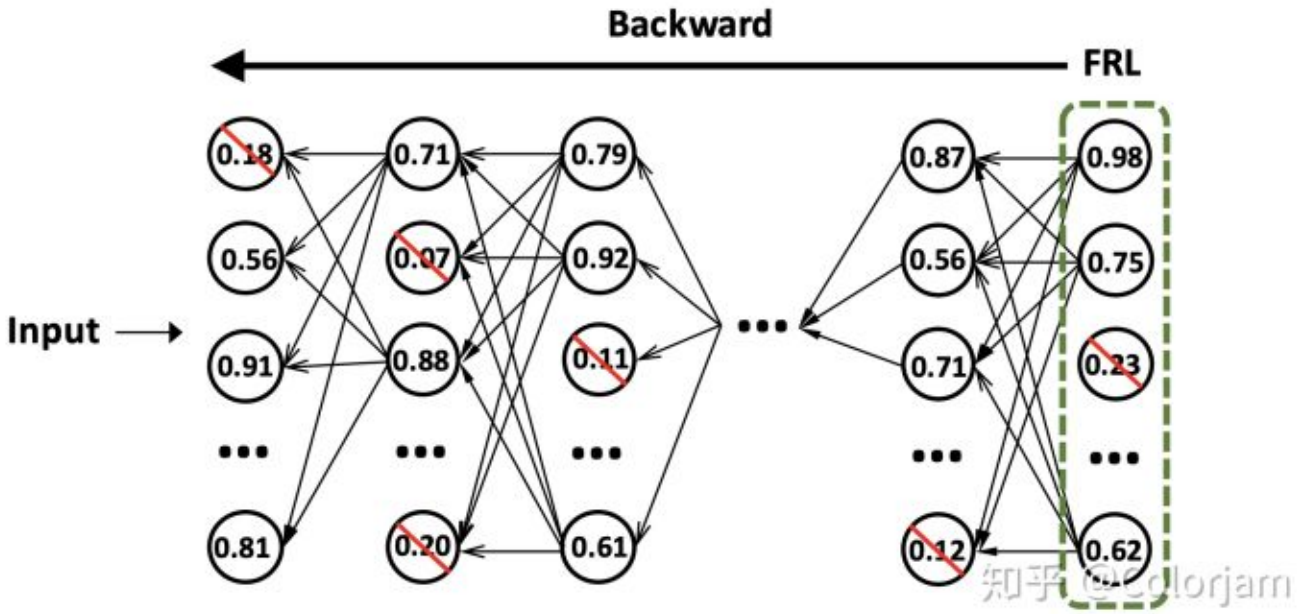
for Neural Network Pruning(CVPR2019)，换成weight的展开再加个平方。类似的方法还有Faster gaze prediction with dense networks and Fisher pruning，用Fisher信息来近似Hessian矩阵。SNIP: Single-shot Network Pruning based on Connection Sensitivity(ICLR2019)则直接利用导数对随机初始化的权重进行非结构化剪枝。相关工作同样可以追溯到上世纪80年代末Skeletonization: A Technique for Trimming the Fat from a Network via Relevance Assessment(NIPS1988)。历史总是惊人的相似：



还有一些考虑实际硬件部署并结合度量标准进行剪枝的方法，对网络层的剪枝顺序进行了选择。Designing Energy-Efficient Convolutional Neural Networks using Energy-Aware Pruning(CVPR2017)利用每层的energy consumption来决定剪枝顺序，NetAdapt: Platform-Aware Neural Network Adaptation for Mobile Applications(ECCV2018)建立了latency的表，利用贪心的方式决定该剪的层。



这类方法通过最小化特征输出的重建误差来确定哪些filters要进行剪裁，即找到当前层对后面的网络层输出没啥影响的信息。ThiNet: A Filter Level Pruning Method for Deep Neural Network Compression 采用贪心法，Channel Pruning for Accelerating Very Deep Neural Networks(ICLR2017) 则采用Lasso regression。NISP: Pruning Networks using Neuron Importance Score Propagation(CVPR2018) 通过最小化网络倒数第二层的重建误差，并将反向传播的误差累积考虑在内，来决定前面哪些filters需要裁剪。



基于稀疏训练的剪枝

这类方法采用训练的方式，结合各种regularizer来让网络的权重变得稀疏，于是可以将接近于0的值剪掉。Learning Structured Sparsity in Deep Neural Networks 用group Lasso进行结构化稀疏，包括filters, channels, filter shapes, depth。 Data-Driven Sparse Structure Selection for Deep Neural Networks(ECCV2018)通过引入可学习的mask，用APG算法来稀疏mask达到结构化剪枝。A Systematic DNN Weight Pruning Framework using Alternating Direction Method of Multipliers(ECCV2018) 的思想类似，用约束优化中的经典算法ADMM来求解。由于每个通道的输出都会经过BN，可以巧妙地直接稀疏BN的scaling factor，比如 Learning Efficient Convolutional Networks through Network Slimming(ICCV2017) 采用L1 regularizer，Rethinking the Smaller-Norm-Less-Informative Assumption in Channel Pruning of Convolution Layers(ICLR2018) 则采用ISTA来进行稀疏。MorphNet: Fast & Simple Resource-Constrained Structure Learning of Deep Networks(CVPR2018) 也是直接利用L1 regularizer，但是结合了MobileNet中的width-multiplier，加上了shink-expand操作，能够更好的满足资源限制。

Random and Rethinking

有采用各种剪枝方法的就有和这些剪枝方法对着干的。Recovering from Random Pruning: On the Plasticity of Deep Convolutional Neural Networks 就表明了度量标准都没啥用，随机赛高。Rethinking the Value of Network Pruning(ICLR2019) 则表示剪枝策略实际上是为了获得网络结构，挑战了传统的 train-prune-finetune的剪枝流程。Pruning from Scratch 则直接用Network Slimming的方法对训练过程中的剪枝结构进行了一波分析，发现直接采用random初始化的网络权重能够获得更丰富的剪枝结构。

走向NAS的自动化剪枝

从AMC: AutoML for Model Compression and Acceleration on Mobile Devices[ECCV2018]开始将强化学习引入剪枝，剪枝的研究开始套上各种Auto的帽子，玩法更是层出不穷。AutoSlim: Towards One-Shot Architecture Search for Channel Numbers先训练出一个slimmable model（类似NAS中的SuperNet Once for All: Train One Network and Specialize it for Efficient Deployment），继而通过贪心的方式逐步对网络进行裁剪。

Network Pruning via Transformable Architecture Search(NIPS2019) 则把NAS可导的一套迁移过来做剪枝。Approximated Oracle Filter Pruning for Destructive CNN Width

各种拿进化来做的也就不提了。

此外，还有基于信息瓶颈的方法Compressing Neural Networks using the Variational Information Bottleneck(ICML2018)，聚类的方法Centripetal SGD for Pruning Very Deep Convolutional Networks with Complicated Structure(CPVR2019)，等等等等等.....

剪枝之外

提升精度： 利用剪枝的方式来提升模型精度，比如DSD: Dense-Sparse-Dense Training for Deep Neural Networks(ICLR2017)利用非结构化剪枝，阶段性的砍掉某些权重再恢复。稀疏训练Sparse Networks from Scratch: Faster Training without Losing Performance在训练过程中保持网络的稀疏率不变，动态调整层间的稀疏率。

动态结构：不同的输入图片可以走网络中的不同结构。BlockDrop: Dynamic Inference Paths in Residual Networks(CVPR2018)引入一个Policy Network，以Block为单位进行选择。Dynamic Channel Pruning: Feature Boosting and Suppression(ICLR2019)引入SEBlock，以Channel为单位进行选择。Improved Techniques for Training Adaptive Deep Networks采用截断式的选择，简单的图片采用靠前的网路层解决，复杂的加入后面得网络层。

总结

一脉梳理下来感觉做纯的剪枝感觉很难了，对比人工设计的结构和准则，NAS出来的模型可以又小巧精度又高，剪枝也逐渐受其影响快、准、狠地寻找结构。这些效果好的结构和权重背后到底还藏着些什么，请勇士们冲吧。

Reference

- 闲话模型压缩之网络剪枝（Network Pruning）篇
- 技术文章配图指南

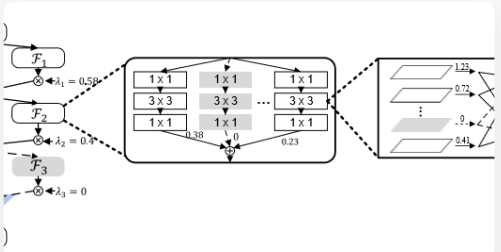
编辑于 02-27

深度学习（Deep Learning） 机器学习 卷积神经网络（CNN）

算法码上来
公众号「算法码上来」分享各种编程及深度学习算法知识

进入专栏

推荐阅读



模型压缩 | 结构性剪枝

孟让

48 条评论

切换为时间排序

写下你的评论...

「已注销」 2019-12-13

这就叫专业.gif

赞

Colorjam (作者) 回复 「已注销」 2019-12-13

向大佬学习.gif

赞

知乎用户 2019-12-13

赞

赞

知乎用户 01-29

专业；另外基于稀疏训练的剪枝这块，ADMM这篇文章也比较经典

赞

秣陵别 回复 知乎用户 02-23

麻烦能给下ADMM那篇链接吗

赞

Colorjam (作者) 回复 知乎用户 02-26

嗯嗯，我添加一下

赞

bestfleer 02-19

Pruning from Scratch这篇我真的非常怀疑结果。。 论文图2里也说了 不同随机权重 剪出来的网络结构相关性都不算高。 结构天差地别。。 那作者怎么就能说 随机的比预训练的好呢。。 而且作者也不公开代码。

赞

Colorjam (作者) 回复 bestfleer 02-23

pruning from scratch复现应该不难，后面可以尝试一下~ 我个人对这篇的理解更倾向于是网络的预训练权重相对于它的结构来说其实不太那么重要（类似rethinking），拿随机的权重来做也能达到比较好的结果。

赞

只是路过看一看 回复 bestfleer 05-29

借助 LOTTERY TICKET HYPOTHESIS 的理论 可以认为初始化后的网络中的部分权重和最后充分训练的权重相差不大（特定lr下），所以可以从随机初始化后的网络进行剪枝操作

赞

秣陵别 02-19

Pruning from Scratch 则直接用Network Slimming的方法对训练过程中的剪枝结构进行了一波分析，发现直接采用random初始化的网络权重能够获得更丰富的剪枝结构。这篇刚讨论

1

Colorjam (作者) 回复 秣陵别 02-23

这点确实有点问题，估计作者是选择了比较好的结果吧。但如果用random做剪枝存在比pre-train好的情况，其实在一定程度上是对剪枝必要性的怀疑了。

赞

秣陵别 回复 Colorjam (作者) 02-26

感觉如果真的好的话，我怀疑我们这批人都要转nas了

赞

展开其他 2 条回复

小米粥 02-25

请问NAS是什么呀，我读论文经常看到，可又查不到[捂脸]

赞

Colorjam (作者) 回复 小米粥 02-26

Neural Architecture Search(NAS) 神经结构搜索，主要研究的是自动生成网络结构。

赞

小米粥 回复 Colorjam (作者) 02-26

嗯嗯，谢谢。

赞

展开其他 2 条回复

hhh77 03-30

所以，总结下来哪个剪枝方法是真的好呢[捂脸]

赞

Colorjam (作者) 回复 hhh77 04-01

这个确实不能下定论（毕竟后出的肯定说比前面的好）但一般全局的效果比层内的好，从简单易用的角度我觉得network slimming挺方便的

赞

hhh77 回复 Colorjam (作者) 04-14

该评论已删除

高大尾巴 04-30

importance estimation for neural network pruning这篇个人感觉方法不是很新？感觉方法和表现上都没有什么创新点。。。（仅个人意见）

赞

Colorjam (作者) 回复 高大尾巴 04-30

确实不是很新（基本和它们17年那篇思路一致），但实验做的还是挺充分的

赞

秣陵别 回复 高大尾巴 05-04

这篇论文是沿着他们之前提的方法做了更多的扩展，比如2阶与1阶进行比较，bn层的影响。实验做了很多，也表明一些结构设计的思路

赞

八八 05-09

感谢楼主专业的分享！由于之前未接触过模型压缩但是有工作需求，在此想问下现在比较成熟的并且落地的（主流）压缩算法或模型框架有哪些？谢谢🙏

赞

Colorjam (作者) 回复 八八 05-11

我了解的目前大厂内部会有自己的压缩框架，开源出来的可以看一下INTEL的Distiller、MSRA的NNi，但其实也不是特别成熟，端到端的通用型压缩方案还是比较缺失的。压缩算法最开始可以用L1等静态剪枝方式尝试一下。

1

八八 05-12

感谢！

赞

Zeyu Zeng 05-19

业务层面其实用处还挺大，感觉现在学术界基本：AutoML is all you need

1

只是路过看一看 05-29

写的很全面了，而且选的论文都很有代表性，大佬称得上此领域专家了。

赞

Colorjam (作者) 回复 只是路过看一看 06-06

不是专家，持续摸索中～

赞

柚柚的天空 06-06

大佬您好，剪枝小白一枚，请教您两个问题，1.目前业界有可以保证精度的裁剪算法嘛？2.是不是获得裁剪结构后的finetune时间都和重新时间基本是一致的呀？求大佬赐教，感激

赞

Colorjam (作者) 回复 柚柚的天空 06-06

不是大佬～1.精度的drop要取决于你的任务、模型、目标稀疏率，如果模型相对于目标任务太冗余，剪枝后精度可能还有提升。但没有算法可以保证精度的不然就没有研究的意义啦。2.我看到的论文设置似乎基本都是一致的。但剪枝后的模型收敛很快，也许可以少一点时间（也和数据集很相关的）。

1

柚柚的天空 回复 Colorjam (作者) 06-10

感谢楼主指导，我看了一下distiller框架，发现好多剪枝算法需要提前指定裁剪层，否则像残差结构这种多依赖关系的层无法自动保证裁剪通道一致，目前有能够自动识别匹配的通用算法嘛？业界都是怎么处理这个问题的呢？

赞

展开其他 3 条回复

vigooo 07-04

楼主您好，想请教一下有什么通道剪枝是能够显著缩短运行时间的，因为最近看的好多策略虽然 flops 降低了很多，但是运行时间并没有显著缩短。是不是这样的方法没有考虑要修剪的每一个模型的结构；又或是什么样的原因？ 请楼主赐教 。。

赞

Colorjam (作者) 回复 vigooo 07-06

模型真实的运行时间和硬件很相关的。一般来说通道数是8的倍数硬件比较友好。有时候通道剪完不是8的倍数，硬件可能会给你补齐通道，那加上一些额外开销自然就没什么加速效果了。

1

做这方面的内容，设备要求高吗

赞

Colorjam (作者) 回复 知乎用户 08-06

在大的数据集和大模型上做比较耗卡

赞

nopro 08-11

想问下大佬，对物体检测的压缩有什么看法嘛？最近有什么好的框架或者算法吗？

赞

Colorjam (作者) 回复 nopro 08-15

物体检测一般backbone用的和分类是一样的吧，剪枝方法可以迁移哒，我似乎没看到专门做检测的压缩。一般比较分类的多，顺带压一下检测模型。

赞

Lycan 09-24

大佬您好，想请教一下有哪些剪枝算法剪枝完的模型，适合硬件平台（NPU等算力平台）上运行呢?

赞

Colorjam (作者) 回复 Lycan 09-24

这个我太清楚～

1

Lycan 回复 Colorjam (作者) 09-24