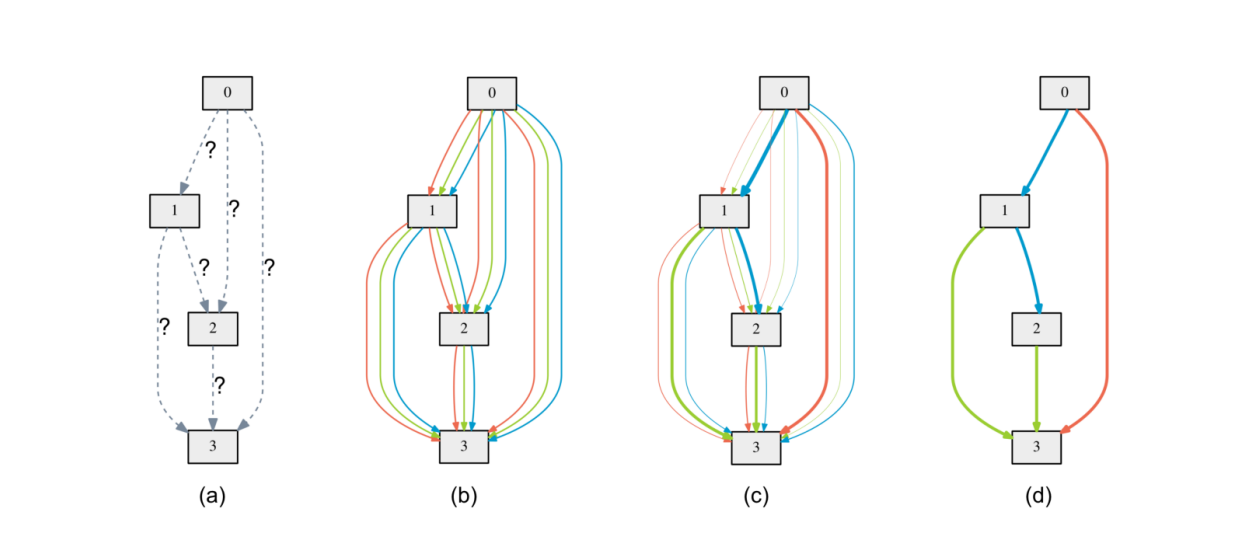
**DARTS：2019 ICLR会议论文**

**阅读原因**：这篇文章可以说是one-short神经网络架构搜索的开山之作，之前在神经网络架构搜索中，由于搜索空间本身是离散型的特点，所以常常是离散化的搜索（基于强化学习或者进化学习的），这篇文章是最早提出连续化搜索的方法之一，尤其是他的开源代码更是对后续的研究起到了很大的促进作用。本人实习期间也是基于DARTS代码中的卷积算子做的工作，所以再重新对其第二次阅读。

**论文内容概述分析**：从摘要中就可以看出（our method is based on the **continuous relaxation** of the architecture representation, allowing efficient search of the architecture using gradient descent.）他篇工作的核心就是这个连续松弛，这个操作开上帝视角看其实就是一个softmax（）的操作，而在神经网络中softmax是可以通过梯度更新的方式进行反向传播的，所以通过这个角度他就将离散化的搜索变成了连续型的搜索。

详细来看DARTS的搜索空间：DARTS神经网络中的每一层就是一个cell算子。为了对照卷积神经网络和循环神经网络。DARTS有两类算子，我称之为卷积算子和循环算子。每一个卷积算子有一个输出，有两个输入，每个输入是前两个cell的输出；每一个循环算子有一个输入和一个输出。由此，搜索整个神经网络架构转变成搜索成ceLL算子即可。图d即为最终的一个算子。

每个算子有两部分构成分别是节点和操作。节点可以理解为卷积神经网络中的特征图，即图a的0，1，2，3。操作即卷积操作，池化，激活函数等等，每两个节点之间有多个候选操作，这就是搜索空间。连续松弛就是在每个候选操作上加一个架构超参数，这个架构超参数经过softmax之后即表示选中这个操作的概率。然后更新参数的方式为两级最优化，在训练集上更新模型本身的参数，在测试集上更新架构参数，最终使其收敛。



**论文优点以及改进方向：**

优点1.提出了基于连续松弛的梯度更新的方式，提出了可微的神经网络结构搜索。且可应用于卷积和循环的神经网路结构。

优点2.在减少gpu训练消耗的前提下，在cifar10上得到了很好的精度

优点3.可迁移到其他的数据集，例如ImageNet并且获得较好的结果。

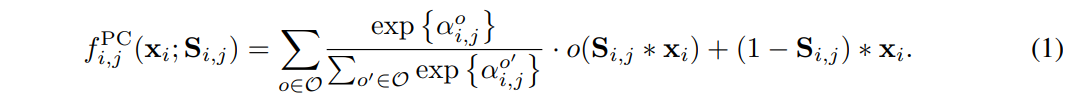
**PC-DARTS: 2020ICLR会议论文**

**阅读原因**：基于DARTS的改进版本，通过这篇论文想了解一下DARTS的某些不足之处。

**DARTS的问题**：结合代码实践和PC-DARTS提出的那样。DARTS其实是一个双阶段搜索加训练的模型，即搜索完模型本身的参数和架构参数后再重新训练。所以它是用cifar搜索出来的神经网络结构，然后再去imagenet数据集上训练。即他只能在小数据集上搜索，如果增大batch\_size后会导致显存GPU不够。

**论文内容概述分析**

为了解决DARTS在搜索过程中显存占用过大和计算量的问题，这篇文中提出了一种思路，即用超网的一部分来进行搜索，具体是每个候选操作的输入都是部分信道。例如每次都是所随机选取有信道的1/4输入卷积，然后最后的结果再与刚开始剩余的3/4相加。即下面公式



但是这样搜索出来的结果会变得不稳定，因为每个特征图的很大一部分来源于之前的特征图，所以模型会更倾向于选择pool，skip而不是卷积。（This regularizes the preference of a weight-free operation (e.g., skip-connect, max-pooling, etc.) over a weight-equipped one (e.g., various kinds of convolution) in O. In the early stage, the search algorithm often prefers weight-free operations, because they do not have weights to train and thus produce more consistent outputs, i.e., o(xi). In contrast, the weight-equipped ones, before their weights are well optimized, would propagate inconsistent information across iterations.），这个会造成准确率的大幅度降低，所以他引入了early normalization的概念，给每个节点的输入来源，即每个比他小的节点乘以权重参数，来保证可以使得经过每个候选操作后特征图差别较大的目的。由于减少了通道，所以可以增加batch\_size进行训练，最终可以直接搜索imagenet的网络结构。

**论文优点以及改进方向：**

1.基于DARTS的论文，对darts搜索过程中显存不足的问题的解决提供了一种思路

2.在imagenet上搜索训练，可以在较短的时间内用较少的资源得到较高的结果

3.感觉减少通道就是让数据一部分不参与训练，这个dropout和调小batch\_size的差别不是特别大，但是准确率确实得到了提高。

**BigNAS: 2020ECCV会议论文**

**阅读原因：**读Attentivenas（2021CVPR）的过程中发现有很多概念不太清楚，这些概念来自于Big NAS，所以对其进行查阅整理，发现其相比于上面连续松弛与池化这个概念，提供了一个新的神经网络结构搜索的概念。即同时训练一个含有很多个子模型（可以达到几千个模型）的大的超网。每个子模型通过蒸馏，参数共享的方式来进行更新。

**论文内容概述分析：**很多神经网络架构搜索都是两阶段的过程，即搜索得到一个模型之后，后续重新还需要对这个模型进行训练。例如darts在cifar10上搜索了一个模型，已经确定了第一层是3\*3卷积，第二层是最大池化，然后是skip。但是为了弥补准确率的差距，往往需要在搜索完成后进行重新训练、微调或其他后处理，但这些步骤会增加计算的复杂度，增加计算成本。BigNAS的提出，让搜索出的模型在不经过额外的fine-tune或者其他后处理操作，也能取得SOTA的效果。

BigNAS的大模型包含很多个小模型，通过改变(kernel and channel sizes, network depths, input resolutions)来决定模型，这些小模型在搜索完之后就可以都达到不错的准确率。那么问题就在于如何得到一个比较好的超网。为此BigNAS提出了一系列的技巧，也可以看做是搜索策略。分别是三明治法则，蒸馏，初始化、学习率调整和正则化（network sampling during training, inplace distillation, network initialization, convergence behavior and regularization）

具体做法是：在每个训练阶段，采样1个最大的子网，1个最小的子网，再随机采样两种大小的网络。在更新整个模型的权重之前，再把所有采样模型的梯度聚合在一起。这样可以提高最小模型的性能下界和最大模型的性能上界。然后输入数据喂入最大的模型，得到预测标签，再将这个标签喂入小模型。由于搜索空间中的所有子网都是残差网络，我们将每个残差块的最后一个BN层的γ参数设置为0，让每个残差块的output初始化为全0张量。将学习率设置跌到一开始的5%就不再变化了。然后正则化只在最大的那个采用出的子模型上实施。

然后文章还有一部分工作，因为他是一个硬件资源感知的nas方法，所以他会考虑再latency，flops，存储量的约束下选择模型，那么BigNAS在搜索完超网之后，需要根据约束条件选择最佳的网络结构。作者提出了粒度从粗到细的采样策略：首先我们找到最有希望的候选网络结构的大概模样，然后在每个感兴趣的框架体系结构附近用较细的粒度采样不同的网络结构。注意的是这一步不需要重新训练，只需要选择超网的权重即可。

**论文优点以及改进方向：**

1.给了我一种全新的视角来选择神经网络架构，即没有连续松弛，而是通过蒸馏，权重共享的方式训练一个包含很多个子模型的超网。

2.确实解决了two-stage重新训练的问题，只需要单独搜索即可。而且提出的三明治法则，大模型小模型怎么同时训练，正则化等很多细节策略有很多启发意义

3.搜索空间较为丰富。

**AttentiveNAS：2021CVPR会议论文**

**阅读原因：**是神经网络结构搜索two-stage方向的论文之一，对训练和搜索阶段做了更进一步的优化，来实现更好的帕累托边界。

**论文内容概述分析：整理总结了两阶段搜索，即先在超网中训练，得到一个准确率很高的超网（超网中的每个子网准确率都很高），然后第二阶段就是搜索（选择），满足在特定限制例如flops的网络。Attentive**在BigNAS上改进了抽样策略，BigNAS是采样一个最大和最小的模型加入到损失函数中的正则。而Attentive选择采样一批，采用BestUp + WorstUp策略，BestUp用于改进当前最好的帕累托前沿，WorstUp用于提升当前表现最差的候选网络。分别提高上限和下限。然后就是提出了两种方法来有效采样最佳或最差的帕累托前沿。为了（减少计算开销）。一是是使用mini-batch损失的性能评估器:对于每个网络结构，在当前mini-batch训练数据上测量的训练损失，用它作为性能度量;二是预训练精度预测器作为性能评估评估器：在验证集上训练准确性预测器;然后，对于每个网络结构，使用精度预测器给出的预测精度作为性能估计。

总结：当前NAS最大的挑战来源于两个方面：一是网络规模非常大，二是计算的代价太大。

对数据集的搜索敏感度。