ShuffleNet V2

Practical Guidelines for Efficient CNN Architecture Design

论文出发点或背景

目前，网络架构设计主要由计算复杂度的间接度量（如FLOPs）测量。然而，直接度量（例如，速度）还取决于诸如存储器访问成本和平台特性的其他因素。因此，这项工作建议评估目标平台上的直接度量，而不仅仅考虑FLOPs。

将 FLOPs 作为衡量计算复杂度的唯一标准是不够的，这样会导致次优设计。

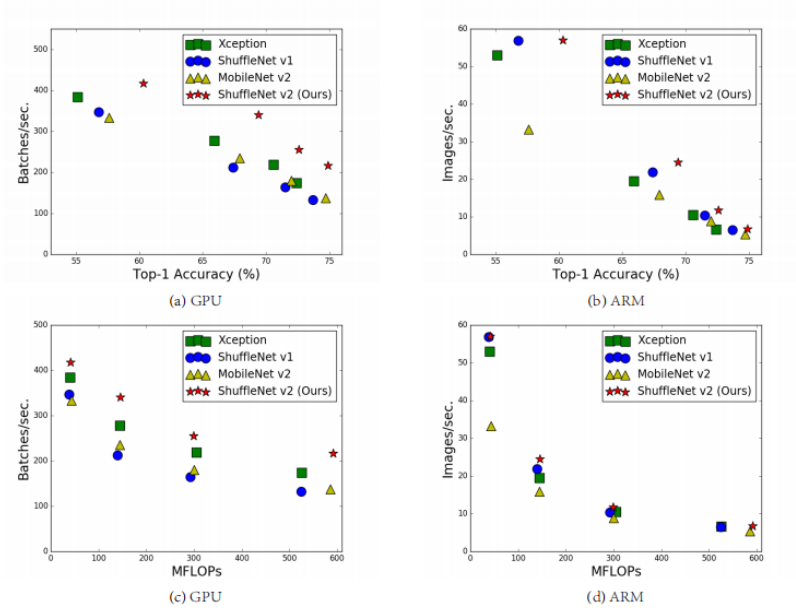

Fig. 1. Measurement of accuracy (ImageNet classification on validation set), speed and FLOPs of four network architectures on two hardware platforms with four different level of computation complexities (see text for details). (a, c) GPU results, $batchsize = 8$. (b, d) ARM results, $batchsize = 1$. The best performing algorithm, our proposed ShuffleNet v2, is on the top right region, under all cases.

间接指标 (FLOPs) 和直接指标（速度）之间存在差异的原因可以归结为两点。首先，对速度有较大影响的几个重要因素对 FLOPs 不产生太大作用。其中一个因素是内存访

问成本 (MAC)。另一个因素是并行度。当 FLOPs 相同时，高并行度的模型可能比低并行度的模型快得多。其次，FLOPs 相同的运算可能有着不同的运行时间，这取决于平台

论文创新思路

高效网络架构设计应该考虑的两个基本原则：

第一，应该用直接指标（例如速度）替换间接指标（例如 FLOPs）；第二，这些指标应该在目标平台上进行评估。

四个实用准则：

G1. 相同的通道宽度可最小化内存访问成本（MAC）；

G2. 过度的组卷积会增加 MAC;

G3. 网络碎片化（例如 GoogLeNet 的多路径结构）会降低并行度;

G4. 元素级运算不可忽视。

所以一个高效的网络结构应该满足：

1）使用「平衡」的卷积（相同的通道宽度，逐点组卷积）；

2）考虑使用组卷积的成本（使用小的分组）；

3）降低碎片化程度（减少并行）；

4）减少元素级运算（捷径连接）。

ShuffleNet V1 严重依赖组卷积（违反 G2）和瓶颈形态的构造块（违反 G1）。

论文方法介绍

本文引入一个简单的操作——通道分割（channel split）

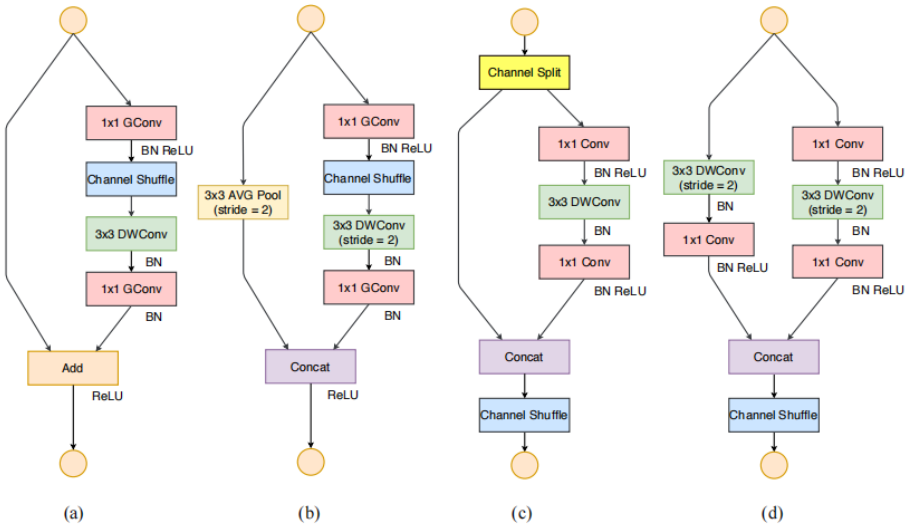

Fig. 3. Building blocks of ShuffleNet v1 [35] and this work. (a): the basic ShuffleNet unit; (b) the ShuffleNet unit for spatial down sampling (2×); (c) our basic unit; (d) our unit for spatial down sampling (2×). **DWConv**: depthwise convolution. **GConv**: group convolution.

在每个单元的开始，c 特征通道的输入被分为两支，分别带有 c−c' 和 c'个通道。按照准则 G3，一个分支仍然保持不变。另一个分支由三个卷积组成，为满足 G1，令输入和输出通道相同。与 ShuffleNet V1 不同的是，两个 1×1 卷积不再是组卷积。这部分是为了遵循 G2，部分是因为分割操作已经产生了两个组。卷积之后，把两个分支拼接起来，从而通道数量保持不变 (G1)。然后进行与 ShuffleNet V1 相同的「Channel Shuffle」操作来保证两个分支间能进行信息交流。

本文提出的构造块 (c)(d)，以及由此而得的网络，被称之为 ShuffleNet V2。基于上述分析，本文得出结论：由于对上述四个准则的遵循，该架构设计异常高效

Layer	Output size	KSize	Stride	Repeat	Output channels			
					0.5×	1×	1.5×	2×
Image	224×224				3	3	3	3
Conv1	112×112	3×3	2	1	24	24	24	24
MaxPool	56×56	3×3	2					
Stage2	28×28		2	1	48	116	176	244
	28×28		1	3				
Stage3	14×14		2	1	96	232	352	488
	14×14		1	7				
Stage4	7×7		2	1	192	464	704	976
	7×7		1	3				
Conv5	7×7	1×1	1	1	1024	1024	1024	2048
GlobalPool	1×1	7×7						
FC					1000	1000	1000	1000
FLOPs					41M	146M	299M	591M
# of Weights					1.4M	2.3M	3.5M	7.4M

Table 5. Overall architecture of ShuffleNet v2, for four different levels of complexities.

实际效果

Model	FLOPs	Top-1 err. (%)
ShuffleNet v2-50 (ours)	**2.3G**	**22.8**
ShuffleNet v1-50 [35] (our impl.)	**2.3G**	25.2
ResNet-50 [5]	3.8G	24.0
SE-ShuffleNet v2-164 (ours, with residual)	**12.7G**	**18.56**
SENet [9]	20.7G	18.68

Table 6. Results of large models. See text for details.

Model	mmAP(%)				GPU Speed (Images/sec.)			
FLOPs	40M	140M	300M	500M	40M	140M	300M	500M
Xception	21.9	29.0	31.3	32.9	178	131	101	83
ShuffleNet v1	20.9	27.0	29.9	32.9	152	85	76	60
MobileNet v2	20.7	24.4	30.0	30.6	146	111	94	72
ShuffleNet v2 (ours)	22.5	29.0	31.8	33.3	**188**	**146**	**109**	**87**
ShuffleNet v2* (ours)	**23.7**	**29.6**	**32.2**	**34.2**	183	138	105	83

Table 7. Performance on COCO object detection. The input image size is 800×1200. *FLOPs* row lists the complexity levels at 224×224 input size. For GPU speed evaluation, the batch size is 4. We do not test ARM because the *PSRoI Pooling* operation needed in [16] is unavailable on ARM currently.

个人理解

ShuffleNet V2我感觉是最近读的轻量化模型里面无论是从方法还是思路最优秀的一篇，相比于之前的轻量级网络都是在追求保持精度不变的情况下尽可能减少网络的参数，但是shufflenet v2开篇指出了之前网络在改进结构的时候考虑不充分，之后通过分析提出了几条设计原则，尤其是G3，多分枝会破坏网络的并行程度，模型速度会越慢.RepVGG也是考虑到了多分枝结构对于推理时的影响，然后利用卷积的可加性进行模型等价，然后加快网络的推理速度。