轻量级网络/检测/分割

由于深度学习的关系，计算机视觉领域发展迅速，各大榜单精度刷的很高，但是这些性能强大的模型却没有用武之地，在现实场景下，更加需要的是轻量级的模型。

本文将从轻量级网络(MobileNetV2、ShuffleNetV2)、轻量级检测(Light-Head R-CNN、ThunderNet)、轻量级分割(BiSeNet、DFANet)3个方面进行介绍。

轻量级网络

好的论文不仅教你为什么，而且教你怎么做，这两篇论文堪称典范，强烈建议多读几遍(就是有点难懂，留下了不学无术的泪水~~)！！！最近ShuffleNetV2还在VALSE上拿了最佳学生论文，膜~~

1.MobileNetV2

Motivation

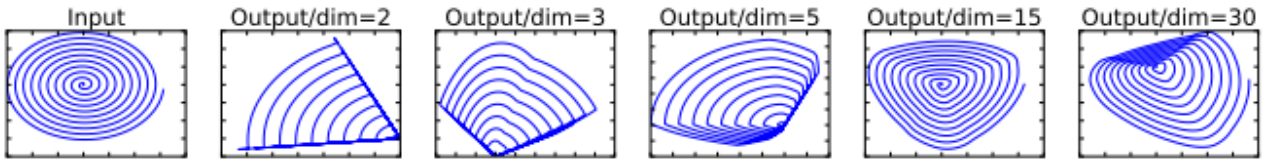


Figure 1: Examples of ReLU transformations of low-dimensional manifolds embedded in higher-dimensional spaces. In these examples the initial spiral is embedded into an n-dimensional space using random matrix T followed by ReLU, and then projected back to the 2D space using T^{-1}. In examples above $n = 2, 3$ result in information loss where certain points of the manifold collapse into each other, while for $n = 15$ to 30 the transformation is highly non-convex.

对一个feature，先通过一个给定的变换规则T，将它映射到它的embedding space中，再在该embedding space中，利用一个ReLU去处理该feature，最后再把这个feature以同样的变换规则（逆方向）给映射回原始空间，这时我们会发现这个feature已经改变了。

维度低的feature，分布到ReLU的激活带上的概率小，因此经过后信息丢失严重，甚至可能完全丢失。而维度高的feature，分布到ReLU的激活带上的概率大，虽然可能也会有信息的部分丢失，但是无伤大雅，大部分的信息仍然得以保留。

由上述可得到以下两条性质：

1.如果the manifold of interest经过ReLU之后信息保持完整，那么该操作实际上退化成了线性变换。

2.如果输入的manifold位于输入空间的低维子空间，那么ReLU能够保留输入manifold的复杂信息。

由以上两条性质可知，the manifold of interest应该位于高维激活空间的低维子空间中。

MobileNetV2引入了两种结构Linear Bottleneck和 Inverted Residual Blocks，既能够去除高维度feature的冗余信息，又能够去除低维度feature的信息坍塌。

Linear Bottleneck

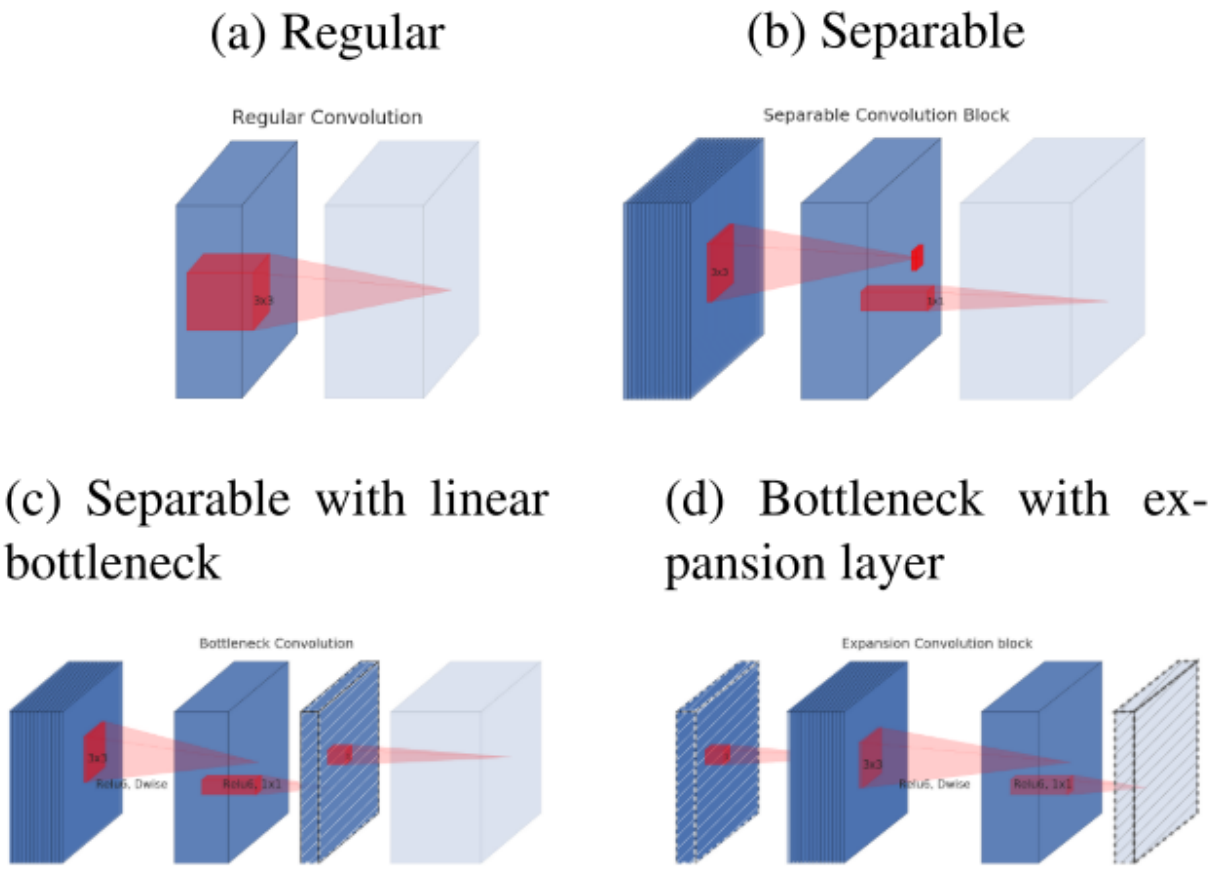


Figure 2: Evolution of separable convolution blocks. The diagonally hatched texture indicates layers that do not contain non-linearities. The last (lightly colored) layer indicates the beginning of the next block. Note: 2d and 2c are equivalent blocks when stacked. Best viewed in color.

与MoblieNetV1的相同点：

都采用了Depth-wise Convolution和Point-wise Convolution组合的方式(Depth-wise Separable Convolution)来提取特征。该操作可以成倍减少时间复杂度和空间复杂度。

与MoblieNetV1的不同点：

1.在Depth-wise Separable Convolution前面添加一个Point-wise Convolution。通过PW操作升维，在高维特征空间中提取特征。

2.去掉Depth-wise Separable Convolution后面的ReLU，等价于一个Linear Bottleneck结构。

由于DW中的PW操作降维，得到一个低维特征空间，由上述分析可知，ReLU会导致低维特征空间坍塌，所以去除PW后面的ReLU可以提升性能。

Inverted Residual Block

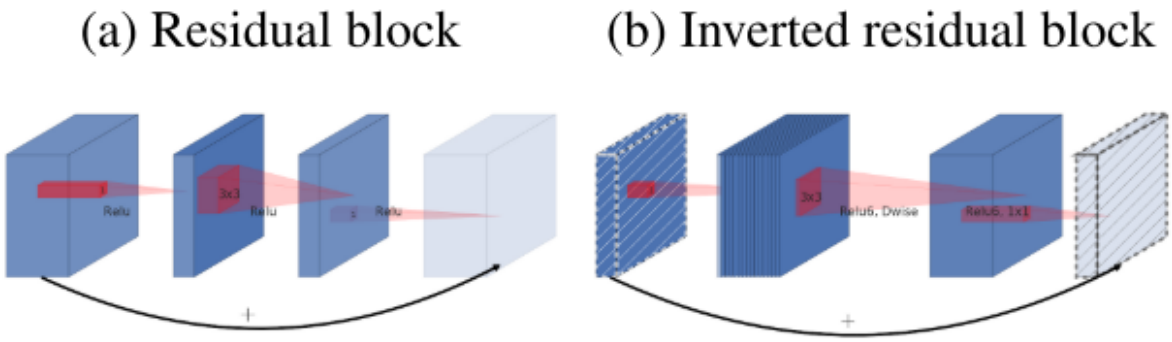


Figure 3: The difference between residual block [8, 30] and inverted residual. Diagonally hatched layers do not use non-linearities. We use thickness of each block to indicate its relative number of channels. Note how classical residuals connects the layers with high number of channels, whereas the inverted residuals connect the bottlenecks. Best viewed in color.

和ResNet的相同点：

1.都采用了1x1 -> 3x3 -> 1x1的模式。

2.都使用了Shortcut操作将输入输出相加。

和ResNet的不同点：

1.ResNet通过标准卷积提取特征，MoblieNetV2通过DW卷积提取特征。

2.ResNet先降维，卷积，再升维，而MoblieNetV2先升维，卷积，再降维(即ResNet是沙漏形，而MobileNetV2是纺锤形)。

通过Linear Bottleneck和 Inverted Residual Blocks，MobileNetV2能够更加充分高效的提取特征。

该部分借鉴了2位大佬的笔记，受益匪浅~~，感兴趣的可以看看大佬的笔记

https://zhuanlan.zhihu.com/p/60668529

https://zhuanlan.zhihu.com/p/33075914

2.ShuffleNetV2

Motivation

以往的工作都是用FLOPs来衡量计算复杂度的，然而FLOPs不是一个直接的指标，我们真正关心的是speed。

两个原则

FLOPs和speed指标之间的差异主要有两个原因：

1.FLOPs指标没有考虑到几个影响速度的重要因素。一个是内存访问成本(MAC)，大部分来自于组卷积。另一个是并行度。

2.有着相同FLOPs的各种操作，在不同平台下的运行时间不同。

基于这些观察，作者提出2个原则：

1.使用直接指标speed。

2.该指标在相同平台下评估。

四个准则

对轻量级网络各种操作的运行时间进行分析，作者提出了4个准则：

G1:通道宽度均衡可以最小化MAC。

G2:增加组卷积会增加MAC。

G3:网络碎片化降低了并行度。

G4:元素操作是不可忽略的。

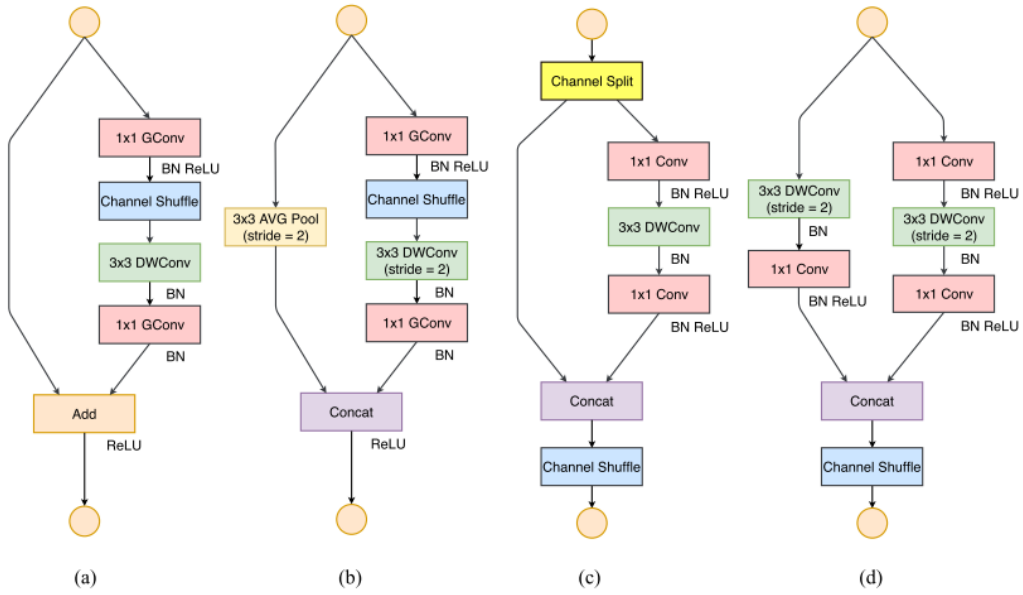


Fig. 3: Building blocks of ShuffleNet v1 [15] and this work. (a): the basic ShuffleNet unit; (b) the ShuffleNet unit for spatial down sampling (2×); (c) our basic unit; (d) our unit for spatial down sampling (2×). **DWConv**: depthwise convolution. **GConv**: group convolution.

ShuffleNetV1

ShuffleNetV1采用了pointwise组卷积和类似bottleneck的结构。另外引入一个channel shuffle操作使得不同组的通道信息流通。

由上述4个准则可知，pointwise组卷积和bottleneck结构增加MAC，这违反了G1和G2。使用过多的组数违反了G3。逐元素相加违反了G4。

因此，为了实现较高的模型容量和效率，关键问题是如何保持大量且同样宽的通道，既没有密集卷积也没有太多的分组。

ShuffleNetV2

在每个单元的开始，输入通道数为c的feature通过channel split分成了2个分支，按照准则G3，一个分支作为identity，另一个分支由3个输入输出通道数相同的卷积组成(满足准则G1)。两个1x1卷积不再是group-wise的了，一部分原因是准则G2，另一部分原因是channel split操作已经分成了两组。卷积后，两个分支concat操作，因此输出通道数和输入通道数保持相同(满足准则G1)。随后引入channel shuffle使得两个分支的通道信息流通。

值得注意的是，ShuffleNetV2去掉了Add操作，元素操作比如ReLU和depth-wise convolutions只在一个分支中存在。并且3个连续的操作Concat、Channel Shuffle和Channel Split被合并成一个元素操作。按照准则G4，这些改变对精度是有帮助的。

对于带有下采样的模块，需要稍微修改一下模块，移除掉channel split操作。输出的通道数增加一倍。

官方解读：

https://zhuanlan.zhihu.com/p/40824527

轻量级检测

大家都知道的One-stage轻量级检测器如YOLO系列、SSD系列就不说了，这里主要介绍一下Two-stage轻量级检测器。

3.Light-Head R-CNN

Motivation

Faster R-CNN和R-FCN在RoI warping之前和之后进行密集计算。Faster R-CNN有2个全连接层，而R-FCN产生大的score maps。这些模型由于heavy-head的设计导致速度变慢。即使显著减小backbone，计算开销还是很大。

作者使用薄的feature map和简单的R-CNN子网络(池化和单个全连接组成)，使得网络的head尽可能的轻。

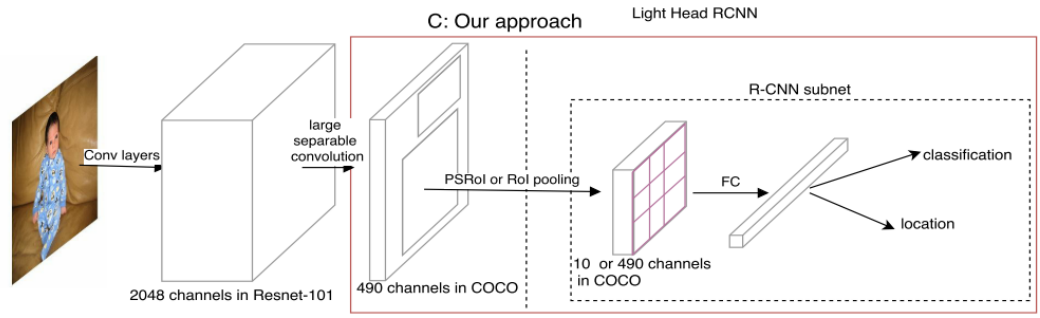


Figure 2. Overview of our approach. Our Light-Head R-CNN builds "thin" feature maps before RoI warping, by large separable convolution. We adopt a single fully-connected layer with 2048 channels in our R-CNN subnet. Thanks for thinner feature maps and cheap R-CNN subnet, the whole network is highly efficient while keeping accuracy.

Basic feature extractor

对于L设置来说，用ResNet来提取特征，对于S设置来说，用Xception-like来提取特征。

Thin feature maps

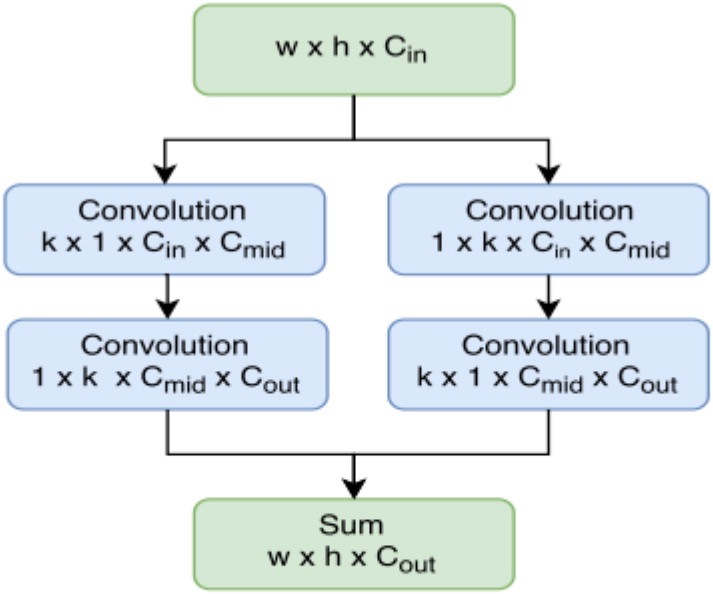


Figure 3. Large separable convolution performs a $k \times 1$ and $1 \times k$ convolution sequentially. The computational complexity can be further controlled through C_{mid}, C_{out}.

作者在C5层后面接一个large separable convolution，k设置为15，对S设置$C_{mid} = 64$，对L设置$C_{mid} = 256$。将C_{out}设置为10 x p x p(R-FCN中为classes x p x p)。large kernel的有效感受野更大，池化后的feature maps的特征会更强。

R-CNN subnet

采用一个2048维的全连接层，后面接2个并行的全连接层来预测分类和回归。

4.ThunderNet

Light-Head R-CNN的做法总的来说还是比较粗糙的，而最近新出的ThunderNet对二阶段检测器进行了全面压缩，从backbone部分到detection部分，从RPN部分到detection head部分，能压缩的全压缩了一遍。不得不说，旷视在轻量级模型设计这一块太厉害了~~ (笔记提到的6篇除了MobileNetV2全是旷视的，膜 ~~)

Motivation

与小的backbone相结合时，Light-Head R-CNN仍然花费大量的计算在detection部分，这导致了弱backbone和强detection部分的不匹配。这种不平衡不仅导致巨大的计算冗余，而且使网络容易过拟合。

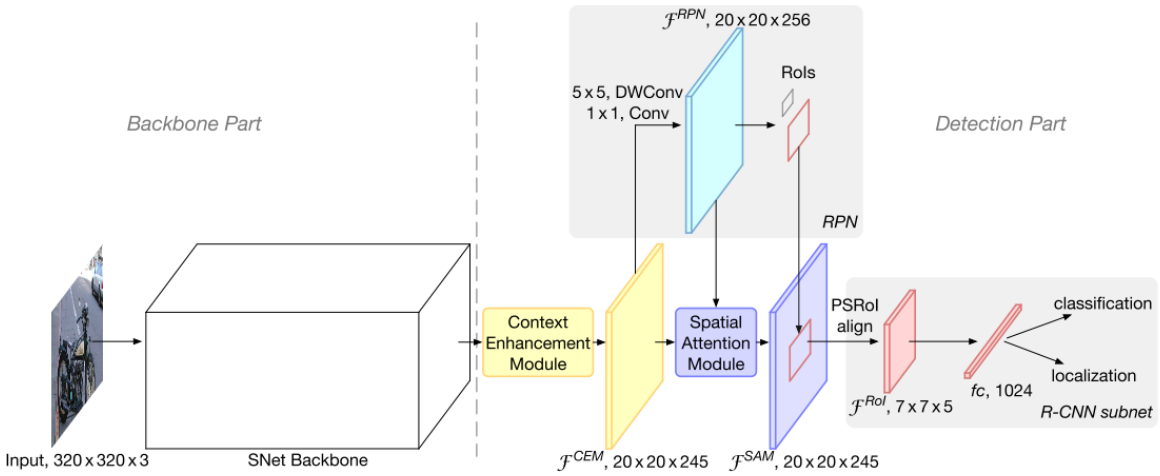


Figure 2. The overall architecture of ThunderNet. ThunderNet uses the input resolution of 320×320 pixels. SNet backbone is based on ShuffleNetV2 and specifically designed for object detection. In the detection part, RPN is compressed, and R-CNN subnet uses a 1024-d *fc* layer for better efficiency. Context Enhancement Module leverages semantic and context information from multiple scales. Spatial Attention Module introduces the information from RPN to refine the feature distribution.

ThunderNet的优化目标是二阶段检测器中计算开销大的结构。在backbone部分，设计了轻量级网络SNet，在detection部分，借鉴Light-Head R-CNN的思路，并进一步压缩RPN和R-CNN子网络。为了避免性能的衰退，设计了2个高效的结构CEM和SAM来改善性能。

Backbone Part 输入分辨率设置为320x320，作者指出输入分辨率应该和backbone的能力相匹配。

作者提出三种SNet的backbones:SNet49推理更快，SNet535性能更好，SNet146速度和性能折中。SNet在ShuffleNetV2的基础上进行了以下两点修改：

1.将ShuffleNetV2中所有3x3 depthwise convolutions换成5x5 depthwise convolutions。

2.SNet146和SNet535中去掉Conv5并且增加前面阶段的通道数。SNet49中将Conv5的通道数压缩成512，并且增加前面阶段的通道数。

Detection Part

Compressing RPN and Detection Head

压缩RPN：用一个5x5 depthwise convolution和一个256通道的1x1卷积来替换256通道的3x3卷积。

压缩Detection Head：Light-Head R-CNN产生5 x p x p的feature map，RoI warping采用PSRoI align。

R-CNN subnet：采用一个1024维的全连接层，后面接2个并行的全连接层来预测分类和回归。

Context Enhancement Module

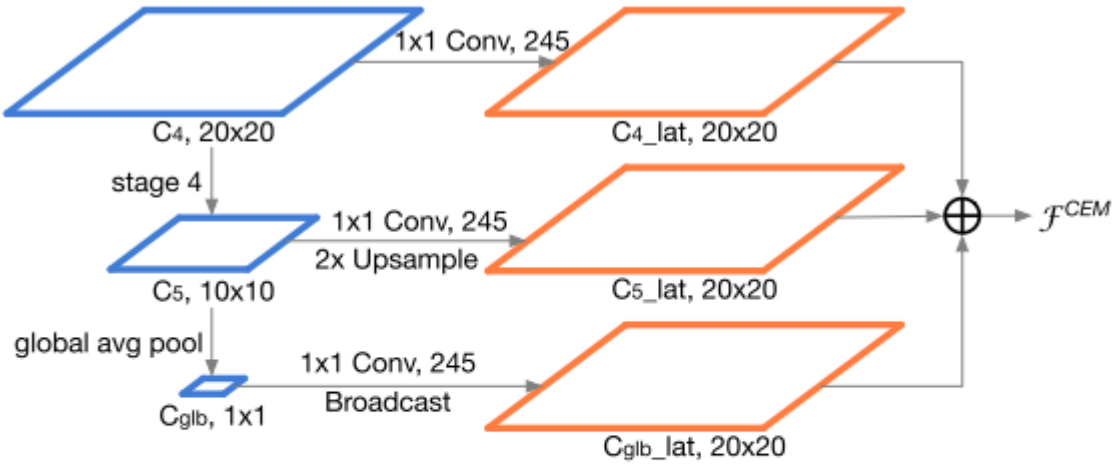


Figure 3. Structure of Context Enhancement Module (CEM). CEM combines feature maps from three scales and encodes more context information. It enlarges the receptive field and generates more discriminative features.

feature maps由三个尺度融合得到，C_{glb}是全局平均池化得到的特征，在每个feature map上使用一个1x1卷积来压缩通道数。C_5进行2倍上采样，C_{glb}进行广播，得到三个分辨率相同的feature map，最后三个feature map进行特征融合。比起FPN结构，CEM只添加了2个卷积和一个fc层，计算开销上更加友好。

Spatial Attention Module

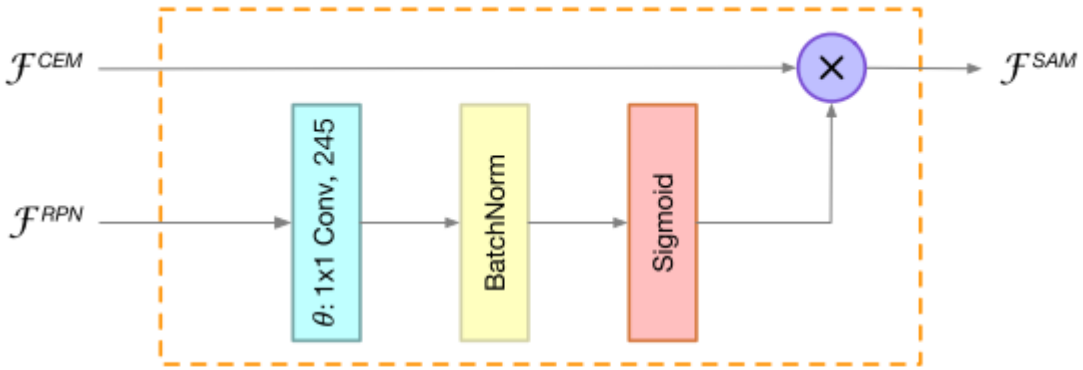


Figure 4. Structure of Spatial Attention Module (SAM). SAM leverages the information learned in RPN to refine the feature distribution of the feature map from Context Enhancement Module. The feature map is then used for RoI warping.

SAM的输入有两个，一个是来自RPN的feature map，另一个是来自CEM的feature map。

输出定义为：

$$\mathcal{F}^{SAM} = \mathcal{F}^{CEM} \cdot sigmoid(\theta(\mathcal{F}^{RPN}))$$

θ转换通道维度，使通道数匹配。

SAM利用RPN的知识来精炼feature map的特征分布，对RoI warping之前的feature map进行权值重标定。

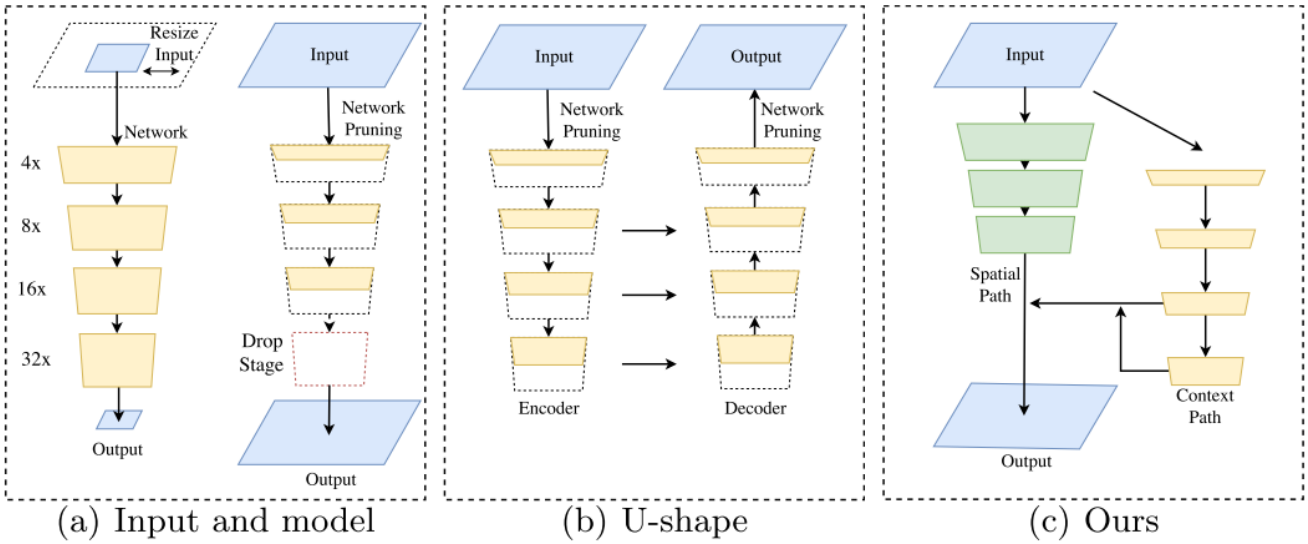
SAM有两个函数，第一个函数，通过增强前景特征并且抑制背景特征来精炼特征分布，第二个函数，来自R-CNN子网络的梯度使得RPN的训练更加稳定。

轻量级分割

以下两篇论文剖析前人工作的部分，分析的非常透彻，很有启发性。

5.BiSeNet

Motivation



实时语义分割主要有三种做法来加速模型(图a)：

1.通过crop或者resize限制输入尺寸，减少计算复杂度。但是这种方法会导致空间细节的丢失。

2.剪枝网络浅层的通道。但是这种方法会减少空间容量。

3.drop模型的最后阶段。但是这种方法丢弃了最后阶段的下采样，导致感受野不足以覆盖大的目标。

为了弥补空间细节的丢失，实时语义分割使用U-shape的结构(图b)。通过融合网络的层级特征，U-shape逐渐增加空间分辨率并且补充了一些缺失的细节。但是，这种做法有两个缺点：

1.因为高分辨率feature maps的额外计算，U-shape结构降低模型的速度。

2.剪枝和裁剪造成的空间信息的丢失是难以恢复的。

Bilateral Segmentation Network

基于对过去工作的分析，作者提出了Bilateral Segmentation Network(图c)，引入了2个组件Spatial Path和Context Path，分别用来解决空间信息的丢失和感受野的不足。为了在速度不降低的情况下提升精度，作者还引入了FFM和ARM结构，分别进行两个分支的融合和精炼预测结果。

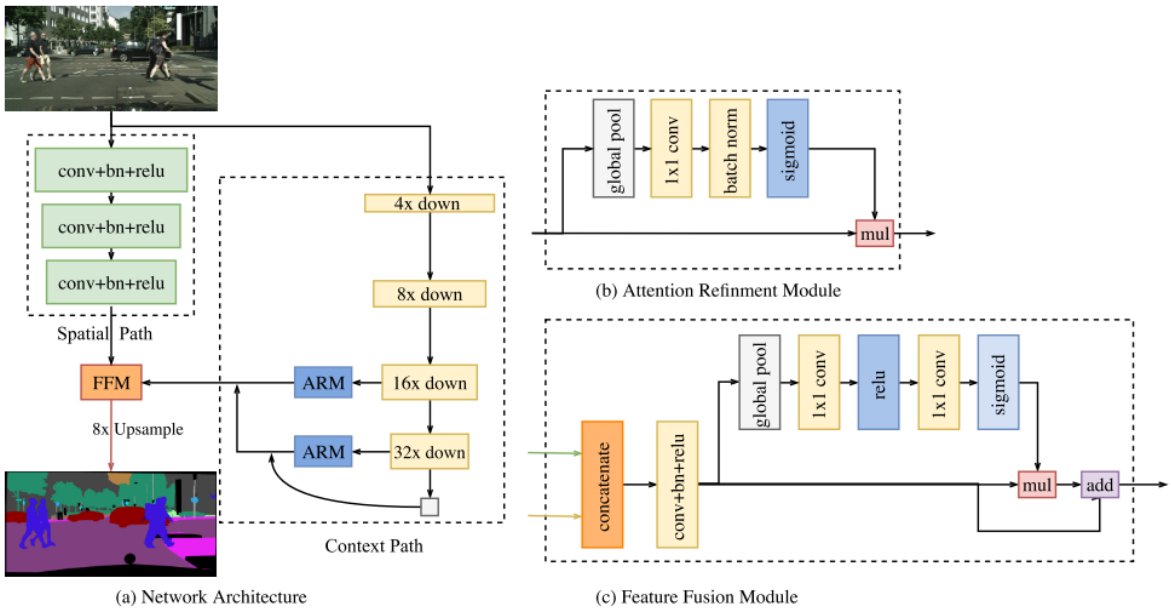


Fig. 2. An overview of the Bilateral Segmentation Network. (a) Network Architecture. The length of block indicates the spatial size, while the thickness represents the number of channels. (b) Components of the Attention Refinement Module (ARM). (c) Components of the Feature Fusion Module (FFM). The read line represents we take this process only when testing.

Spatial path

Spatial path包含3层，每层包含步长为2的卷积、BN和ReLU。分辨率降为输入的1/8。这种编码方式可以保留丰富的空间信息。

Context path

Context path利用轻量级网络和全局平均池化来提供大的感受野。轻量级网络能迅速下采样得到大的感受野，编码高级的语义上下文信息。随后，通过一个全局平均池化，得到最大感受野。最后，将全局池化后的feature上采样和轻量级网络的feature进行融合(最后两个阶段进行U-shape结构融合)。

Attention refinement module

ARM利用全局平均池化来捕获全局信息并且计算得到一个attention向量来引导特征学习。该设计能够精炼Context Path每个阶段的输出特征。

Feature fusion module

因为Spatial Path的输出特征是低级的，而Context Path的输出特征是高级的，所以两个分支的特征不能直接相加融合。

FFM首先将Spatial Path和Context Path分支的输出特征concat，然后利用BN来平衡特征的尺度，最后采用类SENet模块对特征进行权值重标定。

官方解读：

https://zhuanlan.zhihu.com/p/41475332

https://zhuanlan.zhihu.com/p/55263898

6.DFANet

Motivation

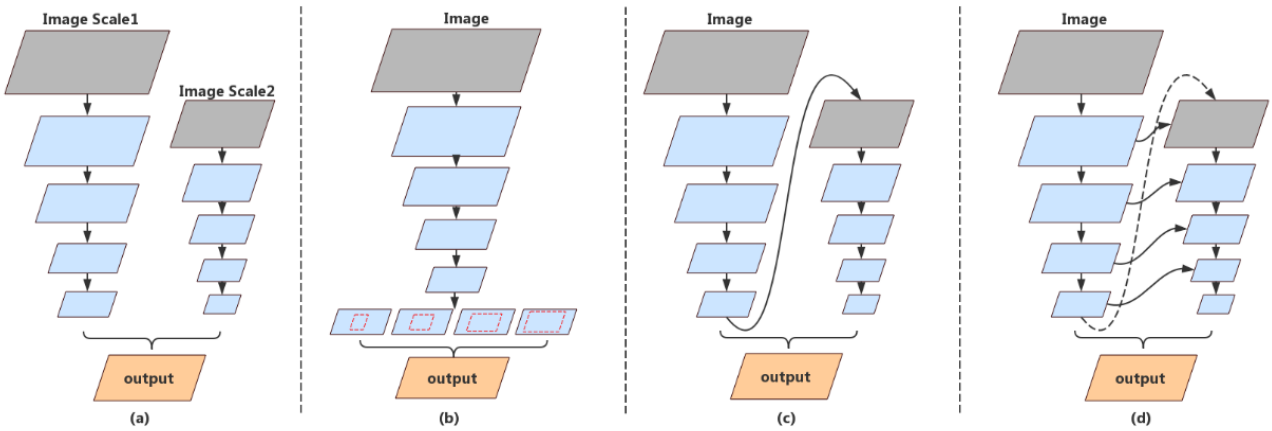


Figure 2. Structure Comparison. From left to right: (a) Multi-branch. (b) Spatial pyramid pooling. (c) Feature reuse in network level. (d) Feature reuse in stage level. As a comparison, the proposed feature reuse methods enrich features with high-level context in another aspect.

实时语义分割的两种方法：

1.使用多个分支来进行多尺度的特征抽取并且保留图像的空间细节(图a)。但是这种方法缺乏对由并行分支结合而成的高级特征的处理，另外，并行分支的特征缺乏信息交流，还有，在高分辨率图像上额外添加分支对速度有影响。

2.使用空间金字塔池化(SPP)模块来处理高级特征(图b)。但是这种方法非常耗时。

受到以上两种方法的启发，作者提出对网络输出进行上采样，然后用另一个子网络对特征进行精炼(图c)。这种做法不同于SPP模块，feature maps在更大的分辨率上进行精炼同时能够学到亚像素的细节。但是这种方法，随着整个结构深度的增长，高维度特征和感受野通常会出现精度损失。

为了进一步提升精度，作者提出stage-level的方法为语义理解提供低级特征和空间信息(图d)。因为所有的子网络有相似的结构，stage-level方法通过conat相同分辨率的层精炼产生多阶段的上下文信息。

Deep Feature Aggregation

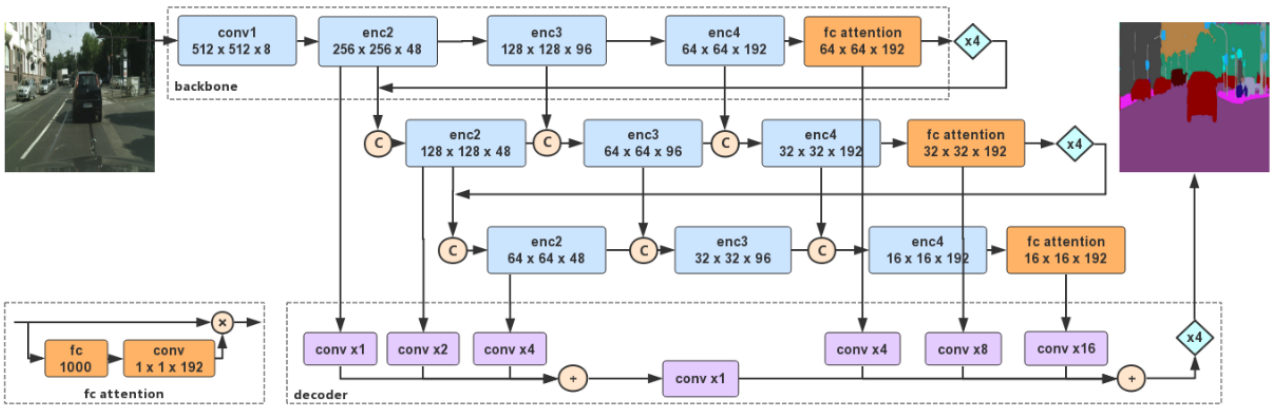


Figure 3. Overview of our Deep Feature Aggregation Network: sub-network aggregation, sub-stage aggregation, and dual-path decoder for multi-level feature fusion. In the figure, "C" means concatenation, "xN" is N× up-sampling operation.

DFANet的特征聚合策略由子网络聚合和子阶段聚合组成。

Sub-network Aggregation

子网络聚合在网络层级进行高级特征的结合。通过将上一个bacebone的输出上采样4倍，然后输入到下一个backbone中，实现子网络聚合。子网络聚合对高级特征进行处理，用以进一步评估和重新评估更高阶的空间关系。

Sub-stage Aggregation

子阶段聚合在阶段层级进行语义合空间信息的融合。在子网络相同深度下对不同阶段进行特征融合。具体的，就是前一个子网络的某个阶段输出是下一个子网络对应阶段的输入。

子阶段聚合公式如下：

$$x_n^i = \begin{cases} x_n^{i-1} + \phi_n^i(x_n^{i-1}) & \text{if } n = 1, \\ [x_n^{i-1}, x_{n-1}^i] + \phi_n^i([x_n^{i-1}, x_{n-1}^i]) & \text{otherwise,} \end{cases} \tag{1}$$

While, x_{n-1}^i is coming from:

$$x_{n-1}^i = x_{n-1}^{i-1} + \phi_{n-1}^i(x_{n-1}^{i-1}) \tag{2}$$

在每个阶段的开始，学习一个残差公式$[x_n^{i-1}, x_{n-1}^i]$。然后使用concat将残差两个分支的特征融合起来。

Network Architecture

DFANet可以看成是一个encoder-decoder的结构。encoder是由3个Xception聚合的，包括子网络聚合和子阶段聚合。decoder简单的设计为特征上采样，然后融合低级和高级特征。所有backbone有相同的结构和相同预训练权重初始值。

Backbone

轻微修改的Xception模型作为backbone。保留来自ImageNet预训练的全连接层来增强语义抽取。在fc层后面使用一个1x1卷积，减少通道数和feature maps匹配。得到的编码向量对输入特征进行权值重标定。

Decoder

首先将3个backbone的深层的高级特征上采样后进行融合。然后将3个backbone的浅层的低级特征上采样后进行融合。最后将融合得到的高级特征和低级特征相加再4倍上采样得到最终的预测结果。

官方解读：

https://zhuanlan.zhihu.com/p/61427477

Reference

1.https://arxiv.org/abs/1801.04381

2.https://arxiv.org/abs/1807.11164

3.https://arxiv.org/abs/1711.07264

4.https://arxiv.org/abs/1903.11752

5.https://arxiv.org/abs/1808.00897

6.https://arxiv.org/abs/1904.02216

欢迎交流指正，感觉写的不错给个赞呗~~ (虽然我也是只收藏不点赞的，哈哈哈哈~~)