CSPNet 论文学习

Abstract

神经网络为计算机视觉任务如目标检测，提供了 state of the art 的方法，取得了难以置信的成绩。但是，这些成绩都过于依赖高计算量，使我们无法在廉价的设备上使用这些先进的方法。本文从网络结构的角度出发，提出了跨阶段局部网络（CSPNet），缓解之前的方法对高计算成本的依赖。作者认为，网络优化过程中存在大量重复的梯度信息。CSPNet 尊重梯度的多样性，将网络中各阶段的特征图整合起来。在ImageNet的实验中，与其它方法相比，CSPNet的计算量降低了20%，而准确率保持不变甚至有所提升。在COCO数据集上，CSPNet的AP_{50}要明显高于其它的SOTA方法。CSPNet很容易实现，也很通用，可以与ResNet, ResNeXt, DenseNet 等方法结合使用。代码位于 https://github.com/WongKinYiu/CrossStagePartialNetworks。

1. Introduction

一般来说，神经网络越深、越宽的话，性能就越好。但是，这样做会带来计算量的显著增加，对于有些任务（如目标检测）来说就比较昂贵。人们越来越需要计算量较低的模型，因为现实环境中的应用程序通常运行在小型设备上，而且推理时间不能长，这给计算机视觉算法带来了不小的挑战。尽管有一些方法是针对移动端CPU设计的，它们所采用的 depth-wise separable convolution 与工业界在边缘计算上的IC设计（如Application-Specific Integrated Circuit, ASIC）不兼容。本文，作者研究了SOTA方法的算力成本，如 ResNet, ResNeXt, DenseNet。作者进一步设计了计算效率比较高的组件，使这些网络可以部署在CPU和移动端GPU上，而不牺牲性能。

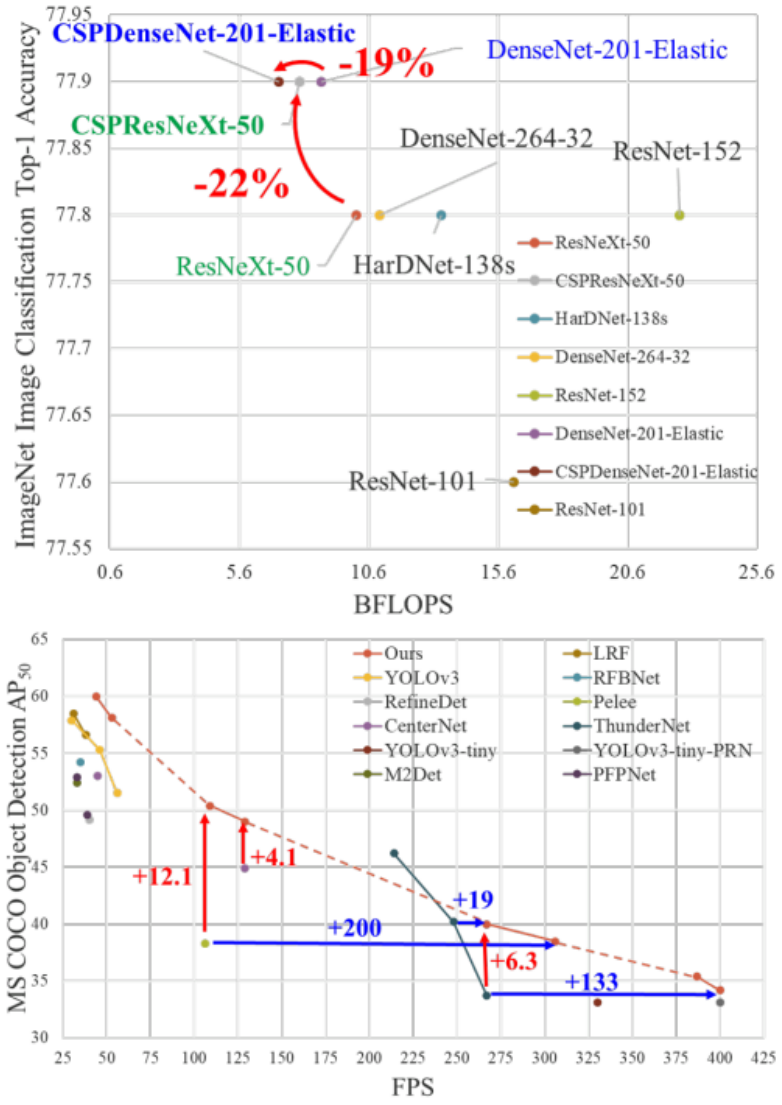

Figure 1: Proposed CSPNet can be applied on ResNet [7], ResNeXt [39], DenseNet [11], etc. It not only reduce computation cost and memory usage of these networks, but also benefit on inference speed and accuracy.

本文中，作者提出了跨阶段局部网络（CSPNet）。CSPNet的设计目的就是让网络在降低计算量的前提下，获取更丰富的梯度融合信息。它将基础层的特征图划分为2个部分，然后再通过一个跨阶段层级将这2个部分融合起来。通过分开梯度流，梯度流就可以在不同的网络路径上传播。这样，通过变换 concat 和 transition 操作，作者发现传播后的梯度流会有较大的相关性差异。此外，CSPNet 能够极大地降低计算量，提升推理速度和准确率，如图1所示。基于CSPNet的目标检测器主要解决了下面3个问题：

- **增强CNN的学习能力**：轻量化后，现有CNN的准确率会大幅度退化，所以作者希望可以增强CNN的学习能力，这样即便轻量化了，也可以保持住准确性。CSPNet 可以轻松地加在 ResNet, ResNeXt, DenseNet 之中，计算量普遍会降低10%到20%，但准确率都超过了本来的算法。
- **去掉算力较高的计算瓶颈结构**：计算瓶颈算力消耗过高，会导致推理时间过长，或者部分运算单元会闲置。因此，作者希望可以将计算量均匀地摊在每一层上，这样就可有效地提升每个计算单元的利用率，降低不必要的资源消耗。CSPNet 将 PeleeNet 的计算瓶颈的计算量几乎降低了一半。在MS COCO数据集上，它将基于YOLOv3的模型的计算瓶颈的算力消耗降低了80%。

- **降低内存占用**：DRAM的晶元制造成本非常昂贵。如果我们可以有效地降低内存成本，我们就可以极大地减少ASIC的成本。为了降低内存使用率，在特征金字塔生成过程中，作者采用了 cross-channel pooling 来压缩特征图。这样，CSPNet 将 PeleeNet 在特征金字塔生成过程中所消耗的内存降低了75%。

CSPNet 可以提升CNN的学习能力，因此就可以用一个较小的模型取得更高的准确率。在 GTX 1080Ti显卡上，CSPNet 在 COCO AP_{50}上取得了50%，速度是109FPS。因此 CSPNet 能够有效地降低显存占用，在 Intel i9-9900K 上，该方法实现了 40%的 COCO AP_{50}，速度是 52FPS。此外，CSPNet 能够极大地降低计算瓶颈，Exact Fusion Model 能够有效地降低所需的内存消耗，在NVIDIA Jetson TX2上，该方法取得了 42%的 COCO AP_{50}，速度是49FPS。

2. Related Works

CNN 结构设计。在ResNeXt中，作者首次提出了 cardinality 比深度和宽度维度更有效。DenseNet 能够显著地降低参数量和计算量，因为采用了大量的复用特征。它将之前所有层的输出特征 concatenate 起来，作为下一层的输入，最大化 cardinality。SparseNet 将 dense 连接调整为指数分布的连接，有效地提高参数利用率，因此效果更好。Wang 等人根据梯度信息结合的想法，进一步解释了高 cardinality 和稀疏连接能够提升网络的学习能力，提出了 partial ResNet（PRN）。为了提升CNN推理的速度，Ma等人在设计 ShuffleNet-V2时提出了4个指导意见。Chao等人提出了低内存占用的CNN，Harmonic DenseNet(HarDNet)，以及一个尺度卷积输入/输出，是对DRAM占用与实际DRAM占用比例的近似。

实时目标检测器：最著名的两个实时目标检测器就是 YOLOv3和SSD。基于SSD，LRF 和RFBNet 在GPU上取得了 state of the art 的实时目标检测性能。最近，anchor-free 目标检测器又逐渐吃香起来，CenterNet 和 CornerNet-Lite 便属于这类检测器，在效率和性能上都不错。对于CPU或移动端GPU上的实时目标检测，基于SSD的 Pelee，基于 YOLOv3的PRN，Light-head RCNN 的 ThunderNet都取得了极佳的表现。

3. Method

3.1 Cross Stage Partial Network

DenseNet：图2 展示了单阶段 DenseNet 的具体结构。每个阶段都包含一个 dense 模块和一个 transition 层，每个 dense 模块由k个 dense 层构成。第i个dense层的输出会与第i个dense层的输入 concat 起来，作为第$i+1$个 dense 层的输入。下面等式就是对上述机制的表示：

$$x_1 = w_1 * x_0$$

$$x_2 = w_2 * [x_0, x_1]$$

...

$$x_k = w_k * [x_0, x_1, ..., x_{k-1}]$$

其中$*$代表卷积操作，$[x_0, x_1, ...]$表示将$x_0, x_1, ...,$ concat 起来，w_i是权重，x_i是第i个 dense 层的输出。

如果我们想要通过反向传播来更新权重，其等式可以表示为：

$$w_1' = f(w_1, g_0)$$

$$w_2' = f(w_2, g_0, g_1)$$

$$w_3' = f(w_3, g_0, g_1, g_2)$$

$$w_k' = f(w_k, g_0, g_1, ..., g_{k-1})$$

其中f是权重更新函数，g_i表示回传到第i个dense 层的梯度。我们可以发现，在更新不同dense层的权重时，大量的梯度信息存在复用的情况。这会使得不同的dense层在不断学习重复的特征。

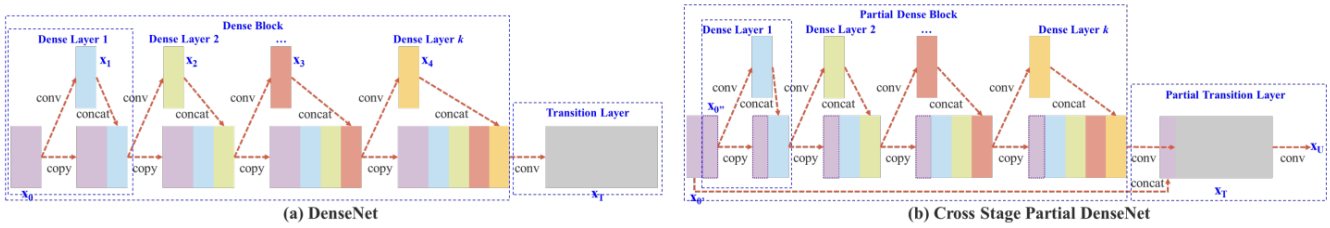

Figure 2: Illustrations of (a) DenseNet and (b) our proposed Cross Stage Partial DenseNet (CSPDenseNet). CSPNet separates feature map of the base layer into two part, one part will go through a dense block and a transition layer; the other one part is then combined with transmitted feature map to the next stage.

Cross Stage Partial DenseNet。单阶段CSPDenseNet的结构如图2（b）所示。CSPDenseNet的一个阶段由一个局部dense模块和一个局部transition层组成。在局部dense模块中，基础层的特征图被分为2个部分，通过通道$x_0 = [x_0', x_0'']$。在x_0'', x_0'中，前者直接与该阶段的末尾连接，后者会穿过整个dense模块。在一个局部transition层中：

- 首先，dense层的输出，$[x_0'', x_1, ..., x_k]$会经过一个 transition层。
- 其次，transition层的输出x_T，会与x_0'' concat起来，然后经过另一个transition层，输出x_U。

前向传播的等式和CSPDenseNet的参数更新如下所示：

$$x_k = w_k * [x_{0''}, x_1, ..., x_{k-1}]$$

$$x_T = w_T * [x_{0''}, x_1, ..., x_k]$$

$$x_U = w_U * [x_{0'}, x_T]$$

$$w'_k = f(w_k, g_{0''}, g_1, g_2, ..., g_{k-1})$$

$$w'_T = f(w_T, g_{0''}, g_1, g_2, ..., g_{k-1})$$

$$w'_U = f(w_U, g_{0'}, ..., g_T)$$

可以发现，一方面来自dense层的梯度会被单独整合起来。另一方面，没有经过dense层的特征图 x'_0 也会被单独地整合起来。要更新权重的梯度信息时，这两部分不会包含对方重复的梯度信息。

总之，CSPDenseNet 保留了DenseNet的优点—特征复用特点，但同时通过截断梯度流，避免了大量的重复的梯度信息。为了实现这个想法，作者设计了一个层级特征融合策略，在局部transition层中使用。

Partial dense block。设计partial dense block的目的是：

- 增加梯度路径：通过拆分与融合策略，梯度路径的数量就增加了一倍。有了cross-stage策略，我们就可以缓解直接复制特征图来 concat 操作所带来的缺点。
- 平衡每层的计算量：通常，DenseNet 基础层的通道个数要远远大于 growth rate。因为partial dense block中 dense层操作的通道个数只是原来的一半，它就可以有效解决接近一半的算力瓶颈。
- 降低内存占用：假设dense block 的基础特征图大小是$w \times h \times c$，growth rate 是d，总共有m个dense层。然后，该dense block的CIO就是$(c \times m) + ((m^2 + m) \times d)/2$，partial dense block的CIO是$((c \times m) + (m^2 + m) \times d)/2$。$m$和$d$通常要远远小于$c$，partial dense block能够节省最多一半的内存占用。

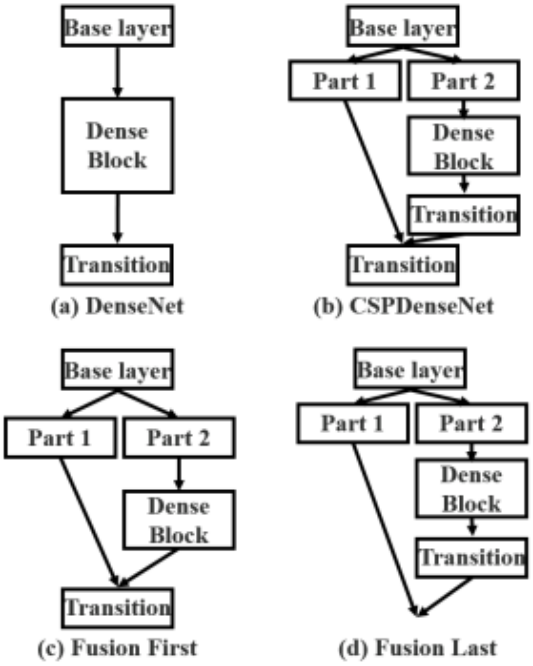

Figure 3: Different kind of feature fusion strategies. (a) single path DenseNet, (b) proposed CSPDenseNet: transition → concatenation → transition, (c) concatenation → transition, and (d) transition → concatenation.

Partial transition layer。设计partial transition layer是为了最大化梯度组合的差异。Partial transition layer是一个层级特征融合机制，使用了梯度流截断的策略来防止不同的层学到重复的梯度信息。作者设计了两个不同的 CSPDenseNet 来介绍梯度流截断策略如何影响网络的性能。图3©和(d)是两个不同的融合策略。CSP（先融合）将两个部分输出的特征图 concat 起来，然后进行 transition 操作。如果采用此策略，会有大量的梯度信息被复用。至于CSP（后融合）策略，dense block 的输出会先通过 transition层，然后与第一部分的特征图进行 concat操作。如果我们采取CSP（后融合）策略，梯度信息就不会被复用，因为梯度流被提前截断了。如果我们用上图中的4个结构来进行图像分类，其各自的结果就如图4所示。我们可以看到，采用CSP（后融合）策略来图像分类，计算量会显著降低，而 top-1准确率只降低了0.1%。另一方面，CSP（先融合）策略会明显地降低计算量，但是top-1准确率会下降1.5%。通过跨阶段的拆分和融合策略，我们就可以在信息整合的过程中，有效地降低冗余的可能性。从图4中可以看到，我们可以有效地降低重复的梯度信息，网络的学习能力得到极大提升。

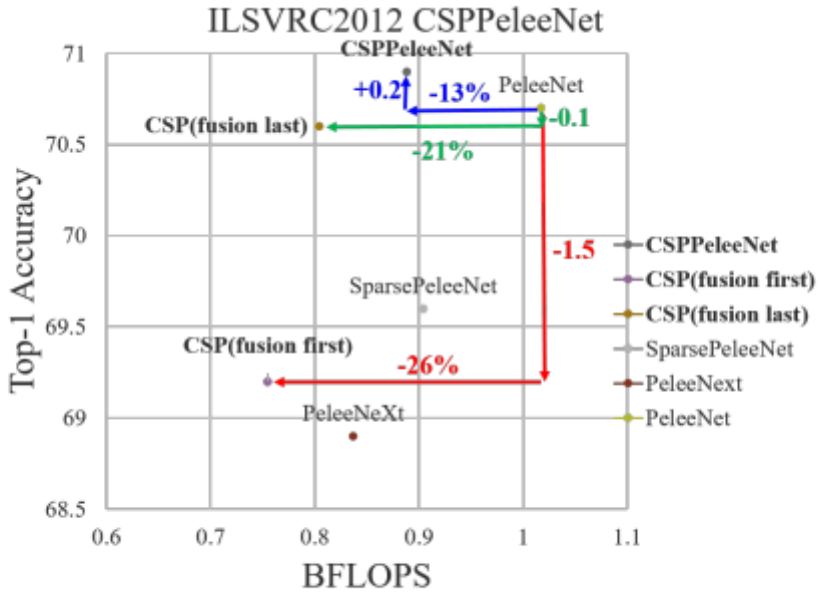

Figure 4: Effect of truncating gradient flow for maximizing difference of gradient combination.

将CSPNet应用在其它网络结构上。CSPNet 也可以轻松地应用在 ResNet和ResNeXt 上，结构如图5所示。由于只有一半的特征通道会经过 Res(X)模块，我们就无需再引入 bottleneck layer。当FLOPs固定的时候，这就是理论上的MAC（memory access cost）下界。

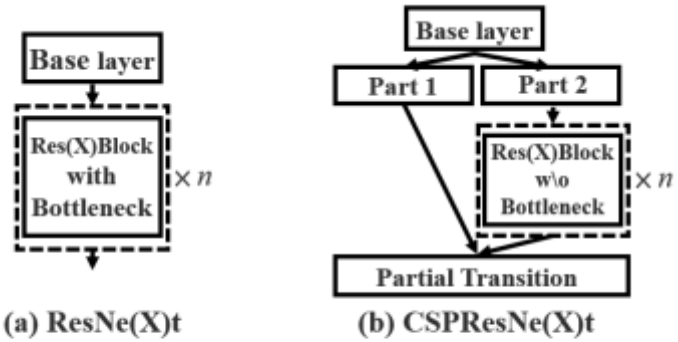

Figure 5: Applying CSPNet to ResNe(X)t.

3.2 Exact Fusion Model

看得准才能预测得准：作者提出了EFM来获取每个anchor的Field of View(FoV)，增强单阶段目标检测器的准确率。对于分割任务，由于像素级的标签通常不包含全局信息，我们更倾向于使用较大的图像区块来提取信息。但是，对于图像分类和目标检测任务，从图像级和边框级的标签上观察，一些重要的信息可能是比较模糊的。Li等人发现CNN学习图像级标签的时候，经常会转移注意力，这就是为什么双阶段检测器要比单阶段检测器效果好。

聚合特征金字塔：EFM能够更好地聚合特征金字塔。EFM基于YOLOv3而来，对于每个 ground truth 物体都有一个边框。每个ground truth边框都对应一个高于阈值IOU的

anchor box。如果anchor box 的大小与网格FoV相等，那么对于第S尺度的网格来说，它里面的边框就是$S-1$尺度的下界与$S+1$尺度的上界。因此，EFM将三个尺度的特征组合了起来。

平衡计算量：因为特征金字塔中concat的特征图非常大，会增加内存占用与计算量。为了解决这个问题，作者提出了Maxout方法来压缩特征图。

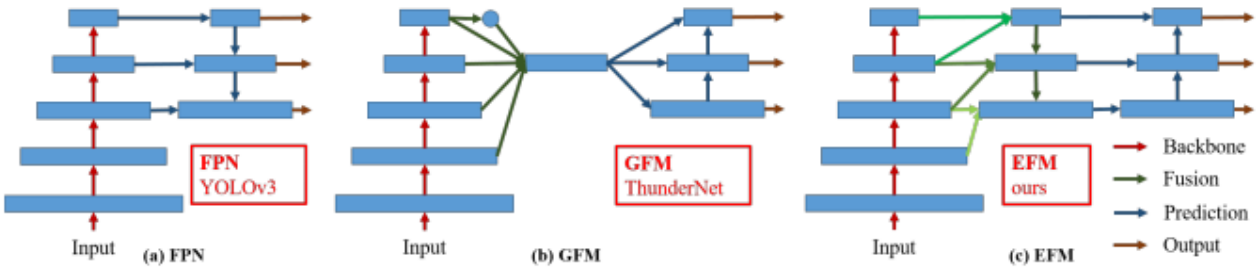

Figure 6: Different feature pyramid fusion strategies. (a) Feature Pyramid Network (FPN): fuse features from current scale and previous scale. (b) Global Fusion Model (GFM): fuse features of all scales. (c) Exact Fusion Model (EFM): fuse features depand on anchor size.

4. Experiments

Pls read paper for more details.