CSPNet：2020

A New Backbone that can Enhance Learning Capability of CNN

论文出发点和背景

通过重新设计架构，缓解网络需要大量推理计算的问题（归咎于网络优化中的重复梯度信息）

通过设计CSPNet能够在减少计算量的同时实现更为丰富的梯度组合。

提出CSPNet主要为了处理以下问题：

1.加强CNN的学习能力

2.消除瓶颈过高的计算瓶颈

3.降低内存成本

论文创新思路

通过设计CSPNet能够在减少计算量的同时实现更为丰富的梯度组合。通过将底层的特征映射划分为两部分，然后通过crossstage hierarchy进行特征融合。

无论是残差网络还是密集网络，跳连会导致将梯度传递到前向的层，进而导致重复学习冗余信息

论文方法介绍

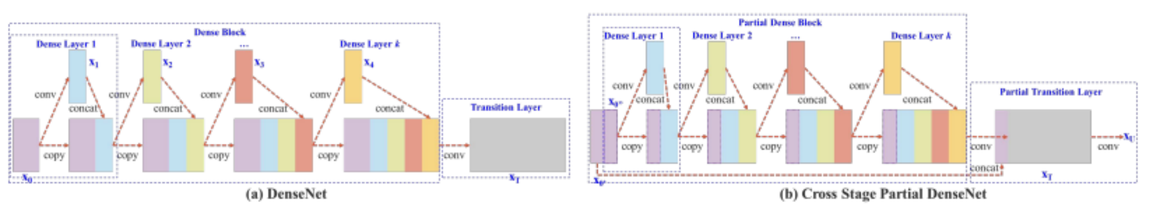

Figure 2: Illustrations of (a) DenseNet and (b) our proposed Cross Stage Partial DenseNet (CSPDenseNet). CSPNet separates feature map of the base layer into two part, one part will go through a dense block and a transition layer; the other one part is then combined with transmitted feature map to the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

传统的卷积神经网络可以被描述为：

$$\begin{aligned} y &= F(x_0) = x_k \\ &= H_k(x_{k-1}, H_{k-1}(x_{k-2}), H_{k-2}(x_{k-3}), \ldots,, H_1(x_0), x_0) \end{aligned} \tag{1}$$

ResNet和DenseNet分别可以被描述为：

$$\begin{aligned} x_k &= R_k(x_{k-1}) + x_{k-1} \\ &= R_k(x_{k-1}) + R_{k-1}(x_{k-2}) + \ldots + R_1(x_0) + x_0 \end{aligned} \tag{2}$$

$$\begin{aligned} x_k &= [D_k(x_{k-1}), x_{k-1}] \\ &= [D_k(x_{k-1}), D_{k-1}(x_{k-2}), \ldots, D_1(x_0), x_0] \end{aligned} \tag{3}$$

x_0被分为了两部分 x0'和x0'',T是transition function用来截断梯度流的函数,M是对两个部分进行融合的函数

图中的Transition Layer代表过渡层，主要包含瓶颈层（1x1卷积）和池化层（可选）

The state-of-the-art methods put their emphasis on optimizing the H_i function at each layer, and we propose that CSPNet directly optimizes the F function as follows:

$$y = M\left(\left[x_{0'}, T\left(F(x_{0''})\right)\right]\right) \tag{4}$$

where x_0 is split into two parts along channel and it can be represented as $x_0 = [x_{0'}, x_{0''}]$. T is the transition function used to truncate the gradient flows of $H_1, H_2, ..., H_k$, and M is the transition function used to mix the two segmented parts. Next, we will show examples of how to integrate CSPNet into DenseNet and explain how to solve the problem of learning duplicate information in CNN.

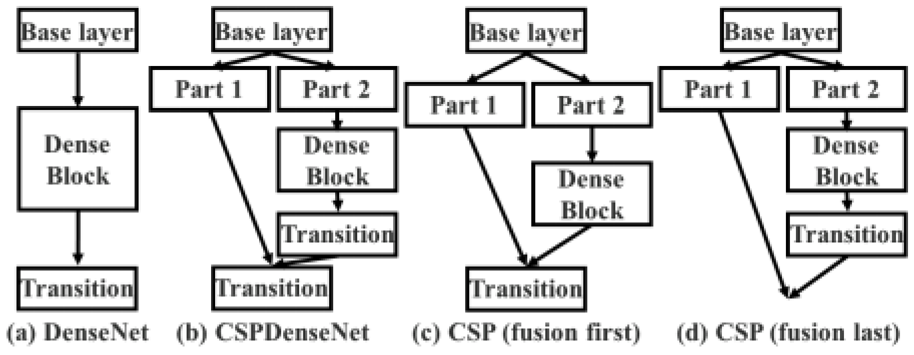

Figure 3: Different kind of feature fusion strategies. (a) single path DenseNet, (b) proposed CSPDenseNet: transition → concatenation → transition, (c) concatenation → transition, and (d) transition → concatenation.

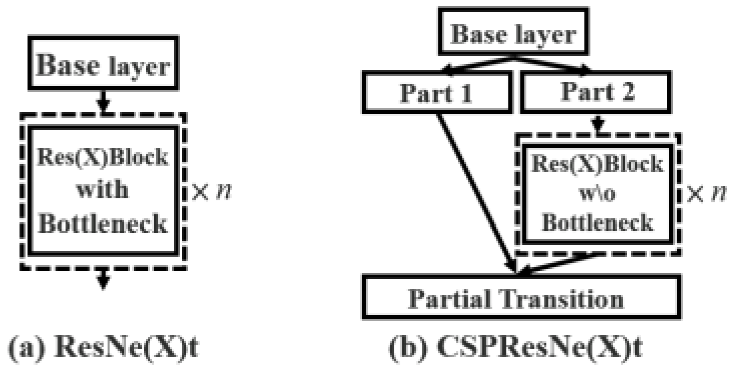

Figure 5: Applying CSPNet to ResNe(X)t.

目标检测检测头：

1.二级检测头优于一级检测头，提出了EFM

2.聚合特征金字塔

3.平衡计算消耗，结合Maxout操作

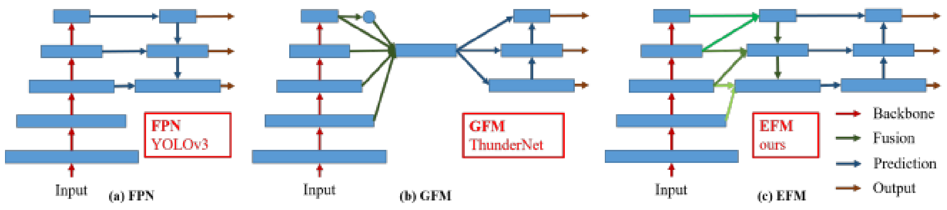

Figure 6: Different feature pyramid fusion strategies. (a) Feature Pyramid Network (FPN): fuse features from current scale and previous scale. (b) Global Fusion Model (GFM): fuse features of all scales. (c) Exact Fusion Model (EFM): fuse features depand on anchor size.

实际效果

将CSP应用到ResNet 、ResNext和DenseNet上后，计算量可以从10%减少到20%，而且在准确率方面还优于上述网络。

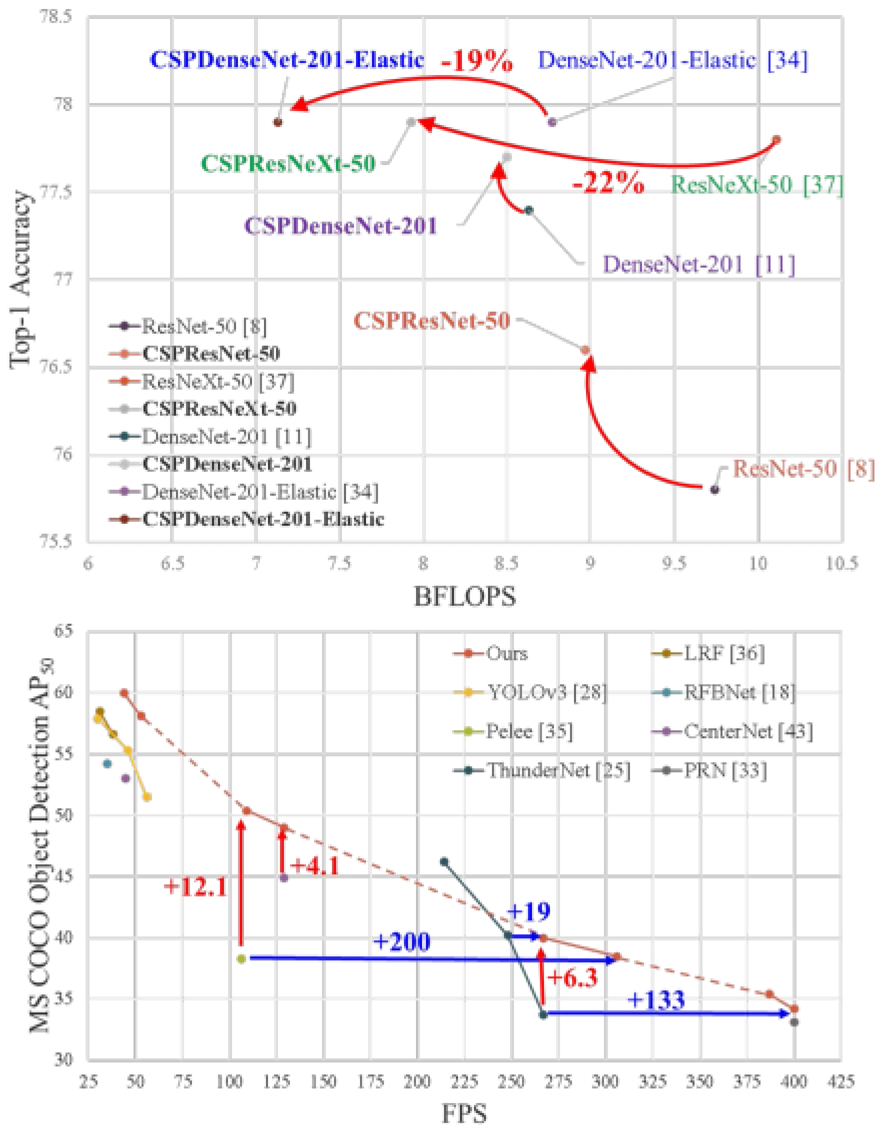

Figure 1: Proposed CSPNet can be applied on ResNet [8], ResNeXt [37], DenseNet [11], etc. It not only reduce computation cost and memory usage of these networks, but also benefit on inference speed and accuracy.

Table 1: Ablation study of CSPNet on ImageNet.

Model	γ	two-way dense	partial dense	trans.	partial trans.	Top-1	BFLOPs
PeleeNet [35]	-	✓		✓		70.7	1.017
	0.75					68.4	0.649
CSP (fusion first)	0.5		✓	✓		69.2	0.755
	0.25					70.0	0.861
	0.75					69.2	0.716
CSP (fusion last)	0.5		✓		✓	70.6	0.804
	0.25					**70.8**	**0.902**
	0.75					70.4	0.800
CSPPeleeNet	0.5		✓		✓	**70.9**	**0.888**
	0.25					**71.5**	**0.986**

Table 2: Ablation study of EFM on MS COCO.

Head	global fusion	exact fusion	atten.	BFLOPs	FPS	AP	AP_{50}	AP_{75}
PRN [33]				3.590	169	23.1	44.5	22.0
PRN-3l [33]				4.586	151	23.7	46.0	22.2
CEM [25]	✓			4.049	148	23.8	45.4	22.6
CEM (SAM) [25]	✓		✓	4.165	144	24.1	46.0	23.1
GFM	✓			4.605	134	24.3	46.2	23.3
EFM		✓		4.868	132	26.4	48.6	26.3
EFM (GIoU [29])		✓		4.868	132	**27.1**	45.9	**28.2**
EFM (SAM)		✓	✓	5.068	129	26.8	**49.0**	26.7
EFM (SPP)		✓		4.863	128	26.2	48.5	25.7

Table 3: Compare with state-of-the-art methods on ImageNet.

Model	#Parameter	BFLOPs	Top-1	Top-5
PeleeNet [35]	2.79M	1.017	70.7%	90.0%
CSPPeleeNet	2.83M	0.888 (**-13%**)	70.9%	90.2%
SparsePeleeNet [44]	2.39M	0.904	69.6%	89.3%
Darknet Reference [26]	7.31M	0.96	61.1%	83.0%
CSPDenseNet Reference	3.48M	0.886	65.7%	86.6%
CSPPeleeNet Reference	4.10M	1.103	**68.9%**	**88.7%**
ResNet-10 [8]	5.24M	2.273	63.5%	85.0%
CSPResNet-10	2.73M	1.905 (**-16%**)	**65.3%**	**86.5%**
ResNeXt-50 [37]	22.19M	10.11	77.8%	**94.2%**
CSPResNeXt-50	20.50M	7.93 (**-22%**)	**77.9%**	94.0%
HarDNet-138s [2]	35.5M	13.4	77.8%	-
DenseNet-264 [11]	27.21M	11.03	77.8%	93.9%
ResNet-152 [8]	60.2M	22.6	77.8%	93.6%
DenseNet-201 [11]	20.01M	8.63	77.4%	93.7%
CSPDenseNet-201	23.07M	8.47(**-2%**)	**77.7%**	**93.6%**
DenseNet-201-Elastic [34]	19.48M	8.77	**77.9%**	**94.0%**
CSPDenseNet-201-Elastic	20.17M	7.13 (**-19%**)	**77.9%**	**94.0%**
Res2NeXt-50 (10 crop) [5]	24.27M	8.4×10	**78.2%**	93.9%
CSPResNeXt-50 (10 crop)	20.50M	7.9×10	**78.2%**	**94.3%**

个人理解

第一次见DenseNet的时候就有一种特征过分重用的感觉，记得在GhostNet里面有张图说是可视化每个通道上的feature，结果发现很多的特征图都是十分相似的。如果像DenseNet那样全部cat过去就会有很多的冗余信息。CSPNet就是通过设计模块，将输入的信息分为两部分，(以改进的DenseNet为例)通过对Dense支路进行先transition后concatenate,阻断一部分梯度信息。CSP不仅吸纳了DenseNet特征重用的优点，同时也没有对重复的梯度信息进行利用。