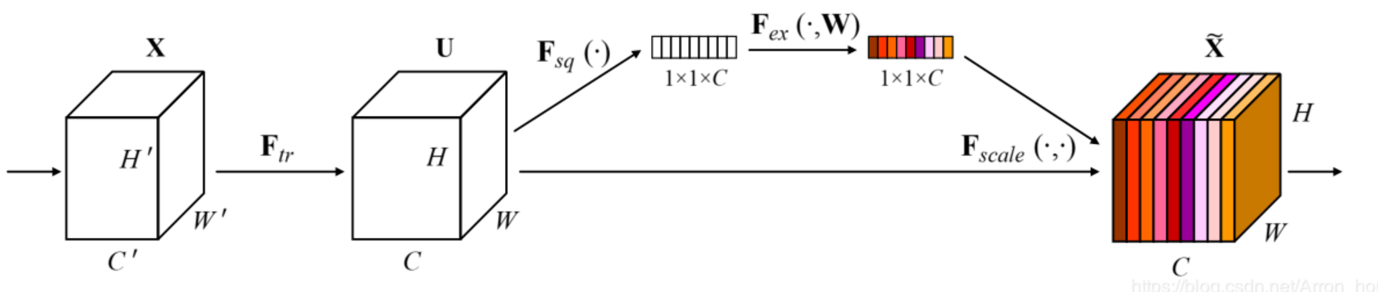
【Attention】注意力机制在图像上的应用

[SeNet] Squeeze-and-Excitation Networks（CVPR2018）

> 论文 https://arxiv.org/abs/1709.01507
> 代码

详细信息可以阅读下面的blog.
SeNet在channel 维进行了信息融合，是一种self_attention的实现。



具体来讲，SeNet是一种卷积结构，将feature map 先压缩，后激励。
压缩Sequeeze
通过global pooling，在channel 维出一个$1\times1\times C$的tensor。然后通过一个bttleneck结构。$1\times1\times C$变为$1\times1\times\frac{C}{r}$通过relu激活后再变为$1\times1\times C$。这个$r$可以是4或者16，4 在imageNet 上的image-top1 err 最低为22.25%，16 在imageNet 上的 image-top1 err 最低为6.03%，backbone 为ResNet-50。

通过bttleneck有两个好处，1是减少模型复杂性，2是增加泛化性。
激励exciation
$1\times1\times C$的tensor最终会通过sigmoid函数，表示每一维channel具有的信息的价值的差异，然后与原始张量channel维相乘。

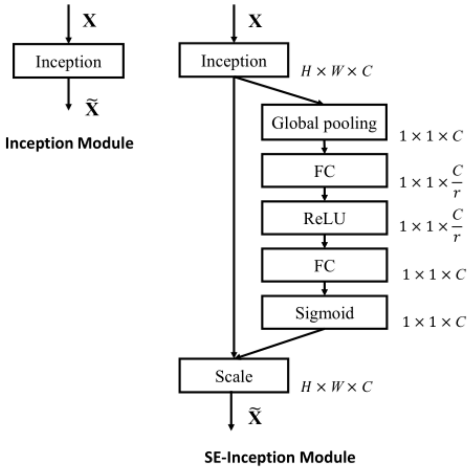


Fig. 2. The schema of the original Inception module (left) and the SE-Inception module (right).

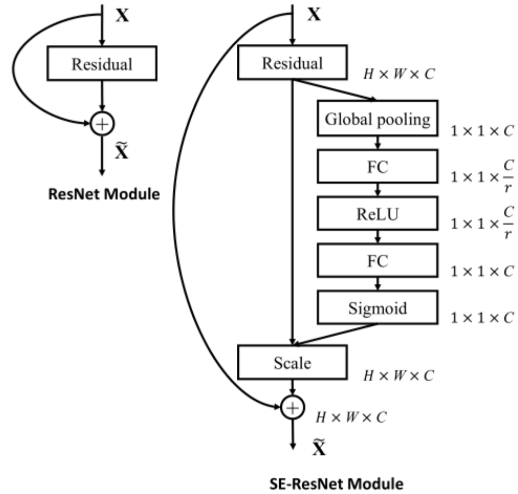


Fig. 3. The schema of the original Residual module (left) and the SE-ResNet module (right).

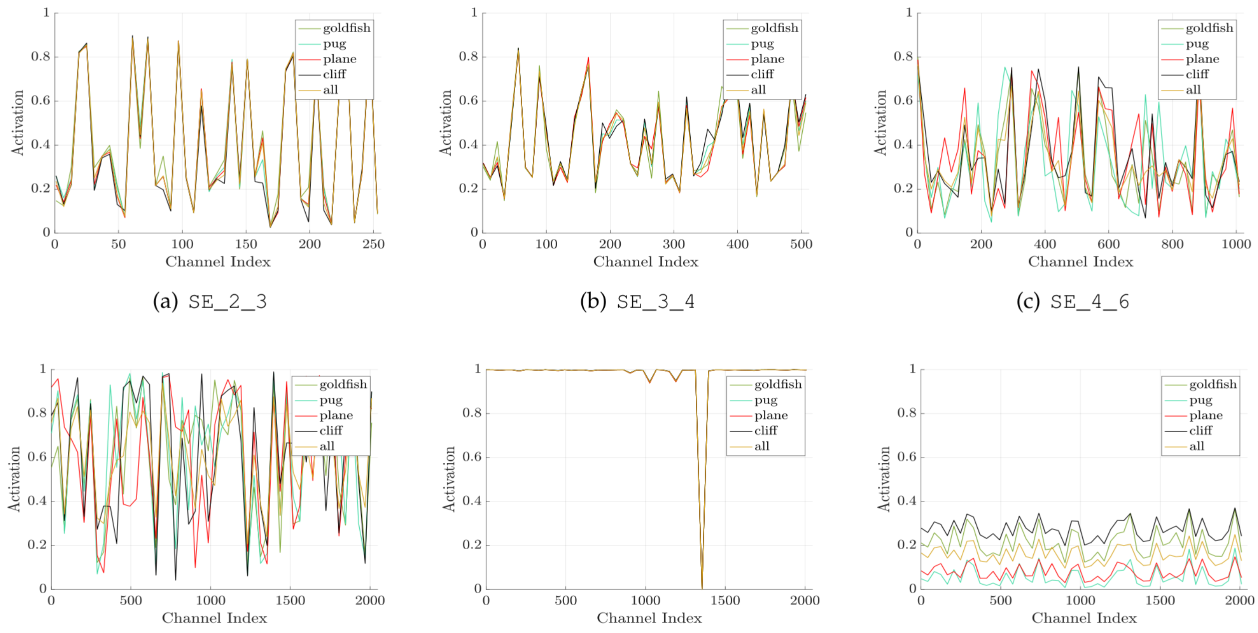
为什么上述的结构会有效呢？
论文中做了简单的分析
在Squeeze 部分，使用Global pooling 起了决定性作用。

Squeeze	NoSqueeze
Global pooling	None
FC1	$1 \times 1 \times \frac{C}{r}$ Conv
FC2	$1 \times 1 \times C$ Conv

TABLE 16
Effect of Squeeze operator on ImageNet (error rates %).

	top-1 err.	top-5 err.	GFLOPs	Params
ResNet-50	23.30	6.55	3.86	25.6M
NoSqueeze	22.93	6.39	4.27	28.1M
SE	**22.28**	**6.03**	3.87	28.1M

使用全局信息会有明显的提升



(a) SE_2_3 (b) SE_3_4 (c) SE_4_6

(d) SE_5_1 (e) SE_5_2 (f) SE_5_3

Fig. 6. Activations induced by the Excitation operator at different depths in the SE-ResNet-50 on ImageNet. Each set of activations is named according to the following scheme: `SE_stageID_blockID`. With the exception of the unusual behaviour at `SE_5_2`, the activations become increasingly class-specific with increasing depth.

1. 在浅层的网络中，不同类别的激活分布十分相似。表示在在此阶段，所有类别共享相同特征通道。早期的特征更具有一般性。
2. 随着网络变深，不同类别所对应的特征通道是不同。较深层的网络逐渐表现出差异性。
3. 在靠经网络输出的层，激励的效果趋向于饱和，表示最后的若干层对于不同类别所提供的差异性不如之前层重要。f 这张图之所以分布类似，偏重不同是因为之后需要输出分类类别。

> https://blog.csdn.net/u014380165/article/details/78006626

[Non-local] Non-local neural Networks （CVPR2018）

> 论文 https://arxiv.org/abs/1711.07971
> 代码 https://github.com/AlexHex7/Non-local_pytorch

Non-local这一篇是在point-wise方面做attention的。

卷积操作的核心在于权重共享和局部相应。通过不断缩小feature

这一篇主要将视频或者音频具有时序顺序的张量的处理。但是思想在2D任然是有效的。可以把所有的T去掉，$1\times1\times1$卷积变为1×1卷积。

具体来说 把一个张量转置后与自身做矩阵相乘。这样每一个像素位置都融合了其他位置的信息。

然后通过softmax激活channel维的张量。

同时，输入张量会通过三个不同的$1 \times 1 \times 1$的卷积层，然后在出口再经过$1 \times 1 \times 1$的卷积层，恢复为原来的channel维数。其实仍然具有一个bottleneck 结构。

最后的结果与原始张量相加。

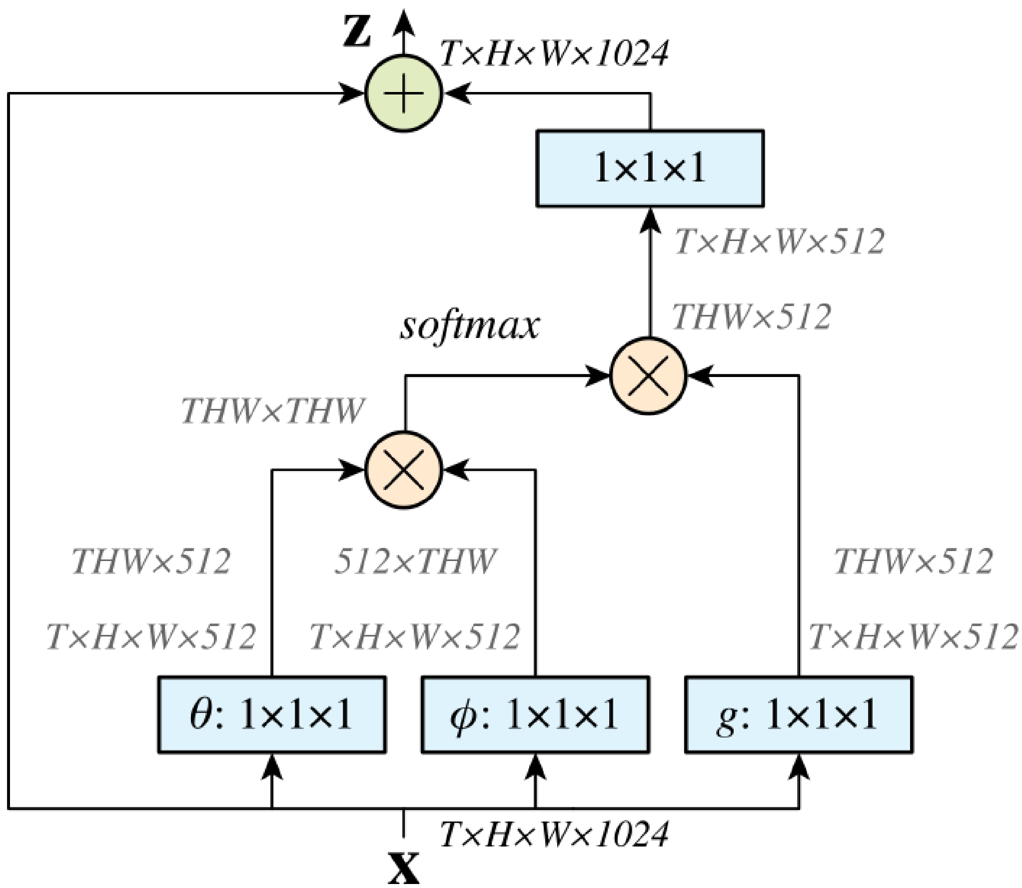


Figure 2. A spacetime **non-local block**. The feature maps are shown as the shape of their tensors, *e.g.*, $T \times H \times W \times 1024$ for 1024 channels (proper reshaping is performed when noted). "$\otimes$" denotes matrix multiplication, and "$\oplus$" denotes element-wise sum. The softmax operation is performed on each row. The blue boxes denote $1 \times 1 \times 1$ convolutions. Here we show the embedded Gaussian version, with a bottleneck of 512 channels. The vanilla Gaussian version can be done by removing θ and ϕ, and the dot-product version can be done by replacing softmax with scaling by $1/N$.

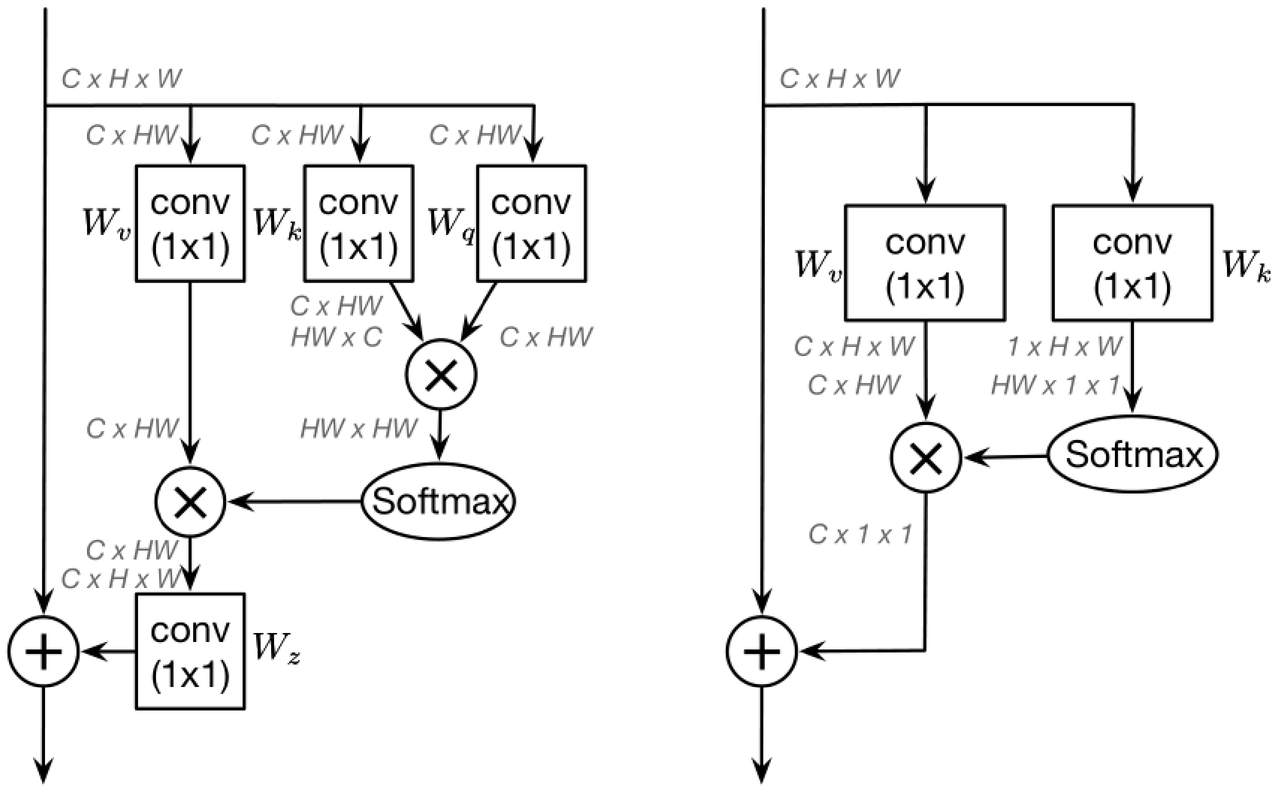
核心代码如下

```
    def forward(self, x):
        '''
        :param x: (b, c, t, h, w)
        :return:
        '''

        batch_size = x.size(0)

        g_x = self.g(x).view(batch_size, self.inter_channels, -1)
        g_x = g_x.permute(0, 2, 1)

        theta_x = self.theta(x).view(batch_size, self.inter_channels, -1)
        theta_x = theta_x.permute(0, 2, 1)
        phi_x = self.phi(x).view(batch_size, self.inter_channels, -1)
        f = torch.matmul(theta_x, phi_x)
        f_div_C = F.softmax(f, dim=-1)

        y = torch.matmul(f_div_C, g_x)
        y = y.permute(0, 2, 1)
        y = y.reshape(batch_size, self.inter_channels, *x.size()[2:])
        W_y = self.W(y)
        z = W_y + x

        return z
```

[GCNet] Non-local Networks Meet Squeeze-Excitation Networks and Beyond 2019-04

> 论文 https://arxiv.org/abs/1904.11492
> 代码 https://github.com/xvjiarui/GCNet

首先对Non-local 做了一个简化。



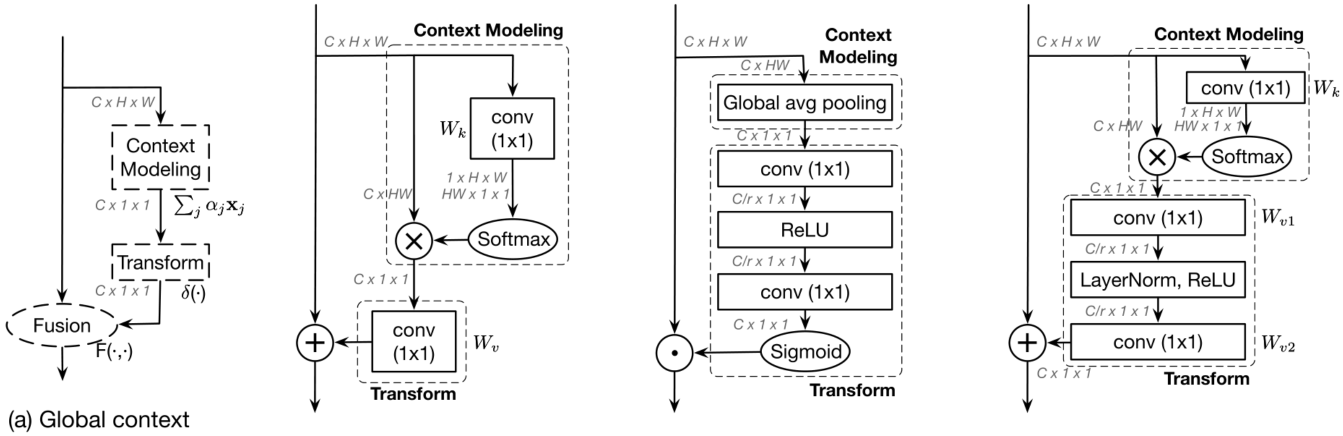
(a) NL block (b) Simplified NL block (Eqn 2)

Figure 3: Architecture of non-local block (Embedded Gaussian) and its simplified version. The feature maps are shown by their dimensions, e.g. CxHxW. $\otimes$ is matrix multiplication, and $\oplus$ is broadcast element-wise addition. For two matrices with different dimensions, broadcast operations first broadcast features in each dimension to match the dimensions of the two matrices.

然后提出一种通用的attention 模块。

将一个attention的过程分为context modeling 以及 transform两部分。

然后取了SeNet的transform部分，simplified Non-local的context modeling部分组成一个新的attention 模块。



(a) Global context modeling framework

(b) Simplified NL block (Eqn 3)

(c) SE block

(d) Global context (GC) block

Figure 4: **Architecture of the main blocks**. The feature maps are shown as feature dimensions, e.g. CxHxW denotes a feature map with channel number C, height H and width W. $\otimes$ denotes matrix multiplication, $\oplus$ denotes broadcast element-wise addition, and $\odot$ denotes broadcast element-wise multiplication.