

想了下，还是要抽时间，把 CV 中的经典模型总结下，其实好多是预训练模型，应该也就是用来提取特征的。

目录

- CV 领域预训练模型 3
 - CNN 经典 3
 - 图像分类 3
 - LeNet-5:用于手写数字识别和英文字母识别 3
 - AlexNet（图像分类） 3
 - VGG-16—图像分类 4
 - Inception Net 5
 - ResNet—残差网络 6
 - Dense Net 6
 - 目标检测 7
 - R-CNN 系列 7
 - 1. R-CNN 7
 - 2. SPPnet (空间金字塔) 8
 - 3.Fast R-CNN 9
 - YOLO 系列 10
 - 激活函数 10
 - 损失函数 11
 - 训练 11
 - GAN 11

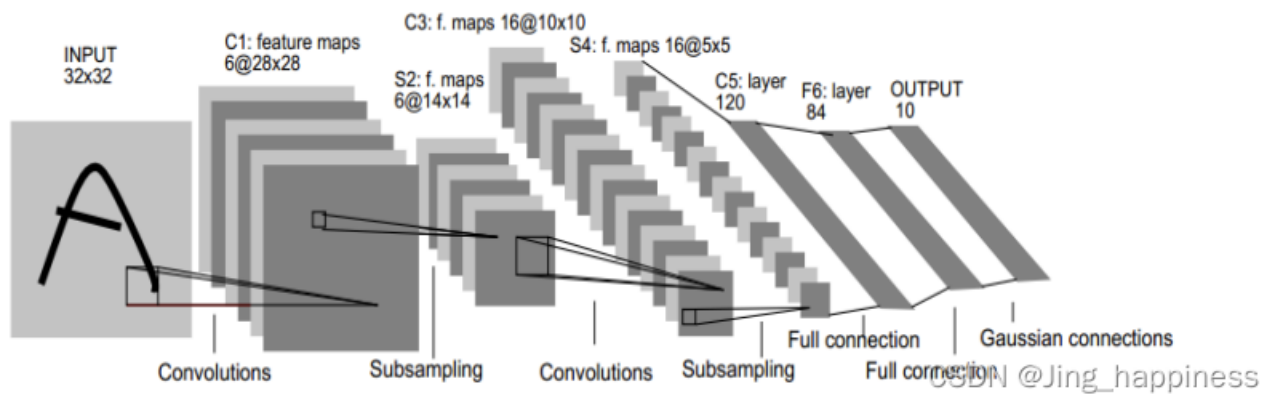
低分辨率到高分辨率图像	11
GP-GAN	11
LAPGAN	13
FCN---图像分割	14
图像质量评测指标	16
PSNR	16
SSIM（结构相似性）：	17
姿势估计评测指标	18
oks	18
AP(Average Precision)平均准确率	18
阅读理解模型	19
R-Net	19
FusionNet	20
全感知融合网络	21
QA-Net	22
阅读理解评估指标	25
阅读理解数据集	25
CNN 中的变形	26
空洞卷积（Dilated convolutions）	27
Pool 的变形	29

CV 领域预训练模型

CNN 经典

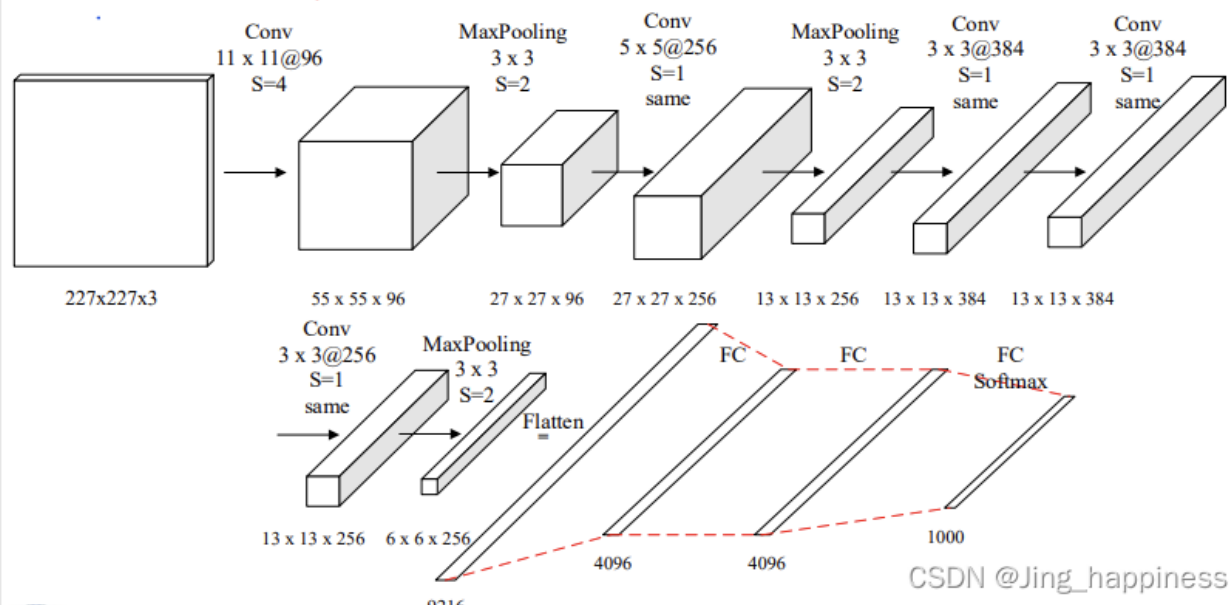
图像分类

LeNet-5:用于手写数字识别和英文字母识别



AlexNet（图像分类）

□ AlexNet可分为8层（池化层未单独算作一层），包括5个卷积层以及3个全连接层



架构描述：

输入层：使用大小为 224x224x3 图像作为输入

第一层：卷积层，包含 96 个大小为 11x11 的卷积核，卷积步长为 4，因此第一层输出大小为 55x55x96；然后构建一个核大小为 3x3、步长为 2 的最大池化层进行数据降采样，进而输出大小为 27x27x96。

第二层：卷积层，包含 256 个大小为 5x5 卷积核，卷积步长为 1，同时利用 padding 保证输出尺寸不变，因此该层输出大小为 27x27x256；然后再次通过核大小为 3x3、步长为 2 的最大池化层进行数据降采样，进而输出大小为 13x13x256

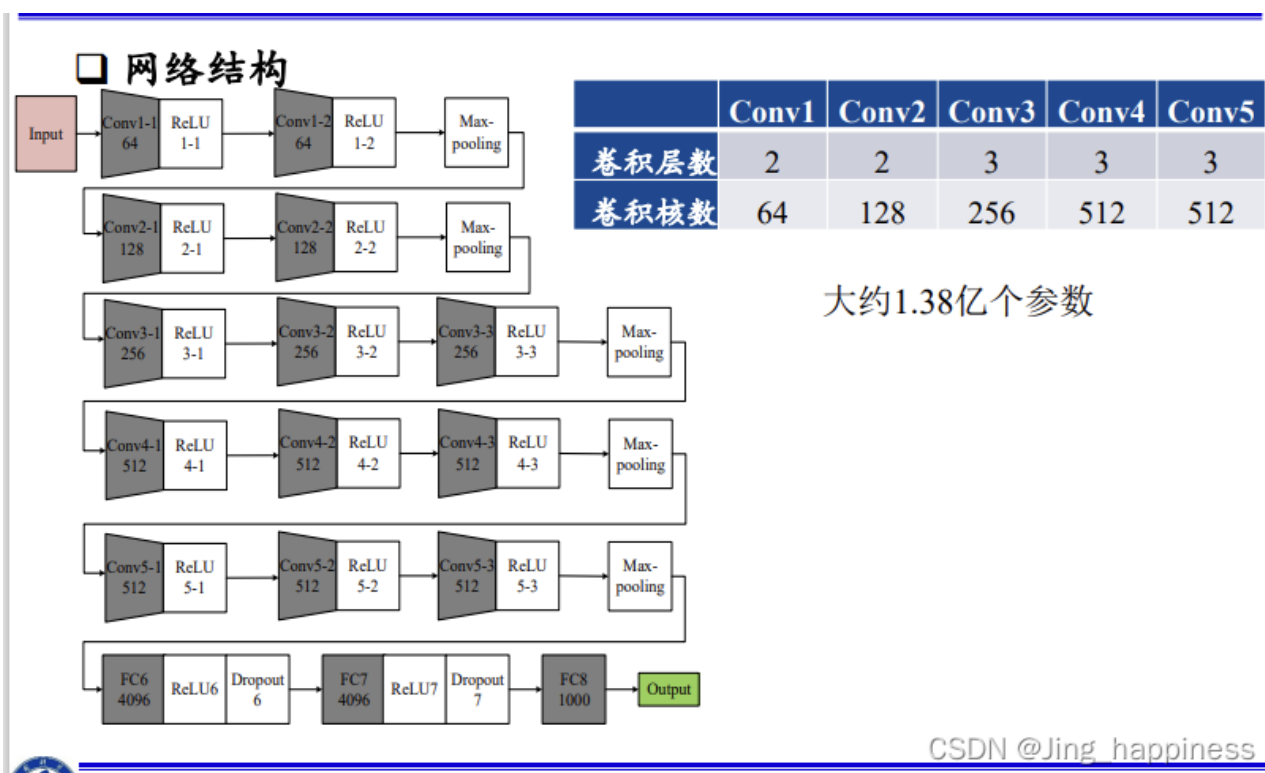
第三层：

第四层：

...

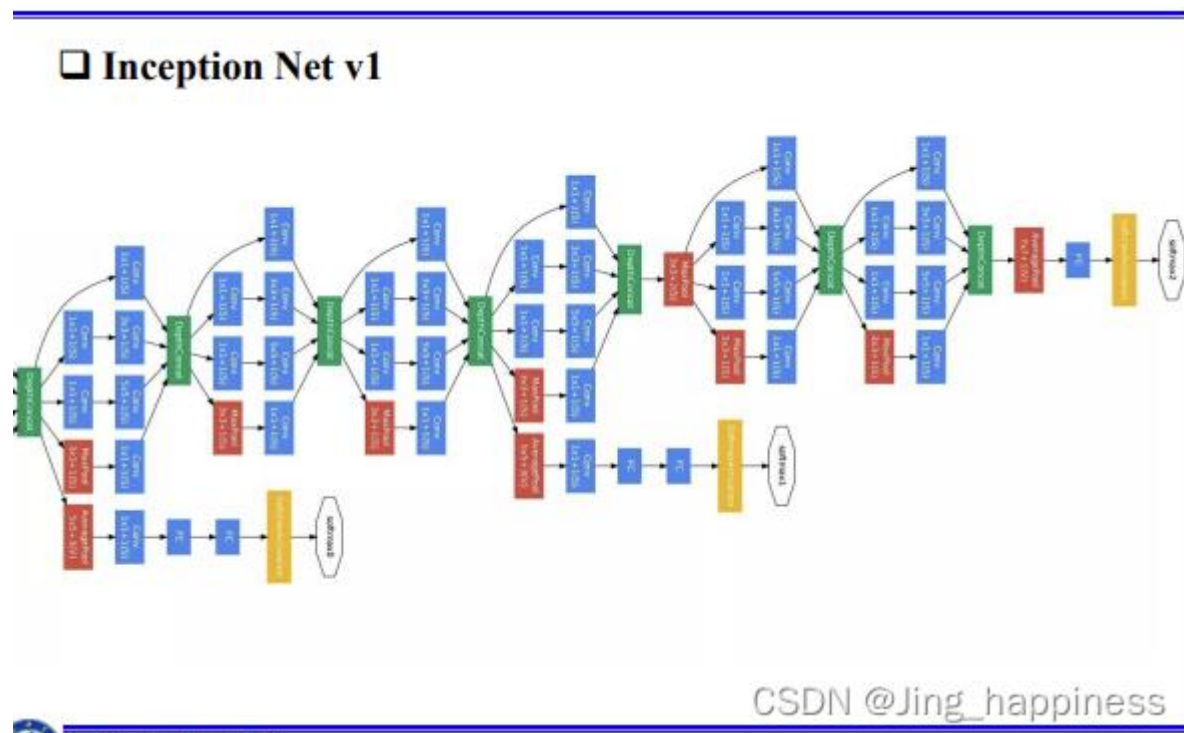
VGG-16-图像分类

相较于 AlexNet，VGG-16 网络模型中的卷积层均使用 3x3 的卷积核，且均为步长为 1 的 same 卷积，池化层均使用 2x2 的池化核，步长为 2。



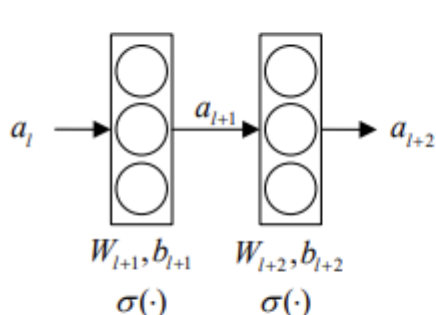
Inception Net

层数更深，采用了 22 层，在不同深度处增加了两个 loss 来避免上述提到的梯度消失问题。

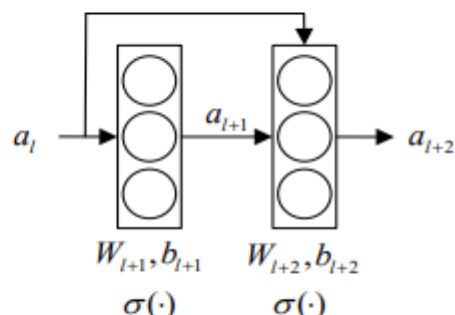


ResNet—残差网络

- ResNet的核心是叫做**残差块** (Residual block) 的小单元, 残差块可以视作在标准神经网络基础上加入了**跳跃连接** (Skip connection)



$$a_{l+1} = \sigma(W_{l+1}a_l + b_{l+1})$$
$$a_{l+2} = \sigma(W_{l+2}a_{l+1} + b_{l+2})$$

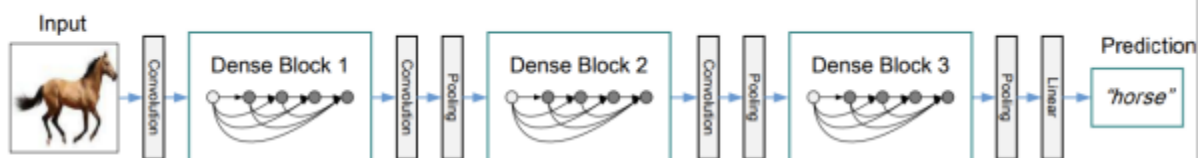


$$a_{l+1} = \sigma(W_{l+1}a_l + b_{l+1})$$
$$a_{l+2} = \sigma(W_{l+2}a_{l+1} + b_{l+2} + a_l)$$

CSDN @Jing_happiness

Dense Net

DenseNet 中, 两个层之间都有直接的连接。对于每一层, 使用前面所有层的特征映射作为输入, 并且使用其自身的特征映射作为所有后续层的输入。



Huang, G., Liu, Z., Van Der Maaten, L., & Weinberger, K. Q. (2017). Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 4790-4798).

CSDN @Jing_happiness

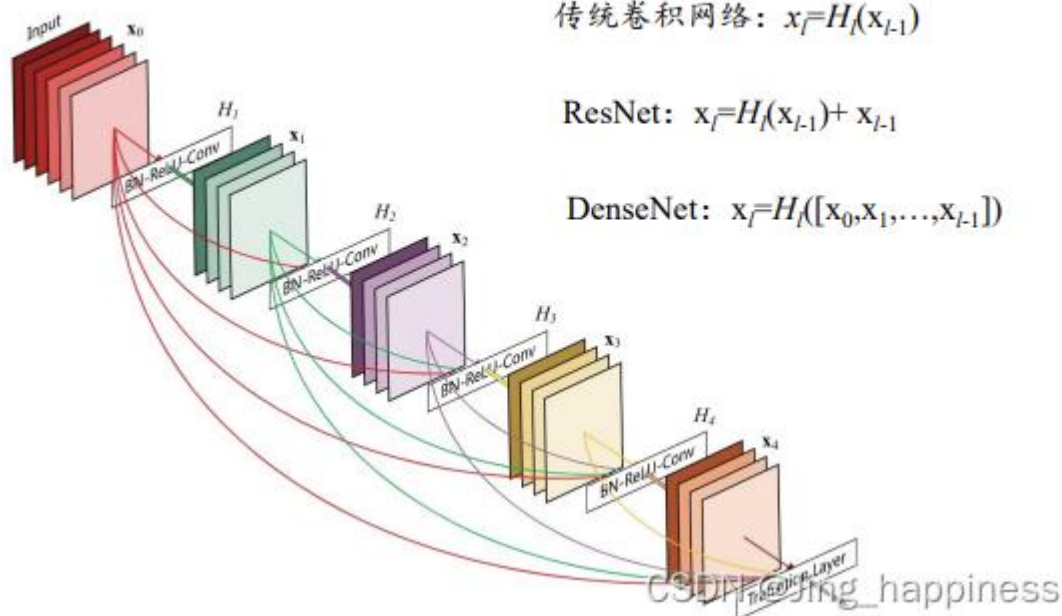
每个 block 的部分如下所示, (设计应该不会用这种结构。在实验中,

DenseNets 随着参数数量的增加, 在精度上产生

一致的提高, 而没有任何性能下降或过拟合的迹象。DenseNet 的优点: -密集连接, 缓解了消失梯度问题- 加强了特征传播, 鼓励特征重用- 一定程度上

减少了参数的数量)

□ 5层的稠密块示意图



目标检测

R-CNN 系列

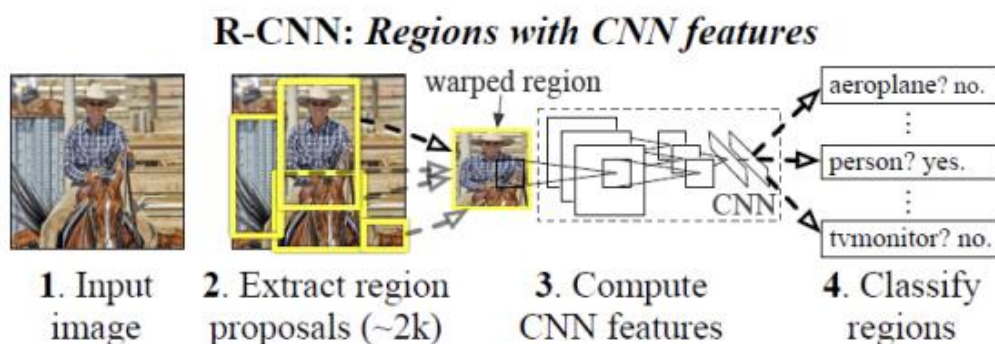
1. R-CNN

1. 图像输入 输入待检测的图像。
2. 区域建议 (Region proposals) 对第一步输入的图像进行区域框的选取。常用的方法是 **Selective Search EdgeBox**，主要是利用图像的边缘、纹理、色彩、颜色变化等信息在图像中选取 **2000** 个可能存在包含物体的区域 (这一步骤 选择可能存在物体的区域，跟分类无关，包含一个物体)。
3. 特征提取 使用 **CNN** 网络对选取的 **2000** 存在物体的潜在区域进行特征提取。但是可能存在一些问题，由于上一步 **Region proposals** 所提取出来的图像的尺寸大小是不一样的，我们需要卷积后输出的特征尺度是一样的，所以要将 **Region proposals** 选取的区域进行一定的**缩放处理 (warped region)** 成统一的 **227x227** 的大小，再送到 **CNN** 中特征提取。R-CNN 特征提取用的网络是对 ImageNet 上的 AlexNet (AlexNet 网络详解) 的 **CNN** 模型进行 **pre-train** (以下有解释，可先行了解 **pre-train**) 得到的基本的网络模型。然后需要对网络进行 **fine-tune**，这时网络结构需要一些修改，因为 AlexNet 是对 1000 个物体分类，fc7 输出为

1000，因此我们需要改为 (class + 1) 若类别数为 20 则应改为 20+1=21 个节点，加一的原因是对图像背景类识别，判断是不是背景。其他的都用 AlexNet 的网络结构 fine-tune（全连接），其中包括五层卷积和两层全连接层。（在这里使用的是 ImageNet 竞赛上面训练好的 AlexNet 模型去除最后两层全连接层的模型（也可以是 VGG，GoogLeNet，ResNet 等）。特征提取用的是卷积神经网络代替了传统的 HOG 特征，Haar 特征等取特征的方法。）

4. **SVM 分类** 将提取出来的特征送入 SVM 分类器得到分类模型，在这里每个类别对应一个 SVM 分类器，如果有 20 个类别，则会有 20SVM 分类器。对于每个类别的分类器只需要判断是不是这个类别的，如果同时多个结果为 **Positive** 则选择概率之最高的。
5. **Bounding Box Regression** 这个回归模型主要是用来修正由第二步 Region proposals 得到的图像区域。同第四步的分类一样，每个类别对应一个 Regression 模型。这个 Bounding Box Regression 主要是为了精准定位。

❑ Object detection system overview



Ross B. Girshick, Jeff Donahue, Trevor Darrell, Jitendra Malik, Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. CVPR 2014: 580-587

2. SPPnet (空间金字塔)

把 SVM、Bbox 回归和 CNN 阶段一起训练，最后一层的 Softmax 换成两个：一个是对区域的分类 Softmax，另一个是对 Bounding box 的微调

– R-CNN主要问题：每个Proposal独立提取CNN features，分步训练

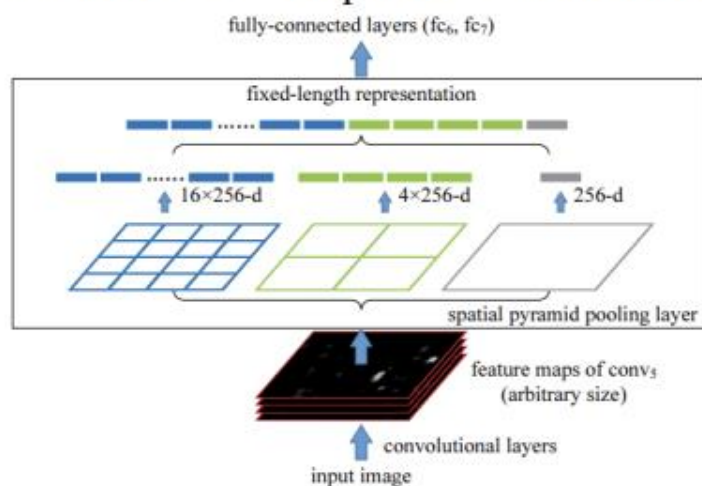


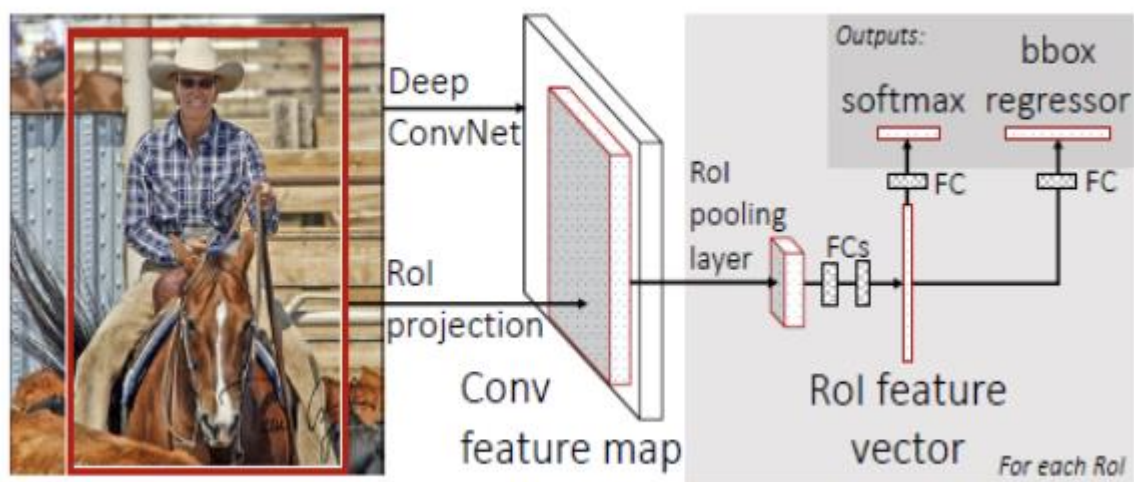
Figure 3: A network structure with a **spatial pyramid pooling layer**. Here 256 is the filter number of the conv₅ layer, and conv₅ is the last convolutional layer.

CSDN @Jing_happiness

3.Fast R-CNN

ROI Pooling 层：将每个候选区域均匀分成 $M \times N$ 块，对每块进行 max pooling，将特征图上大小不一的候选区域转变为大小统一的数据，送入下一层。最后，分别通过 softmax 做分类任务和 regressor 做回归任务。

□ 概述

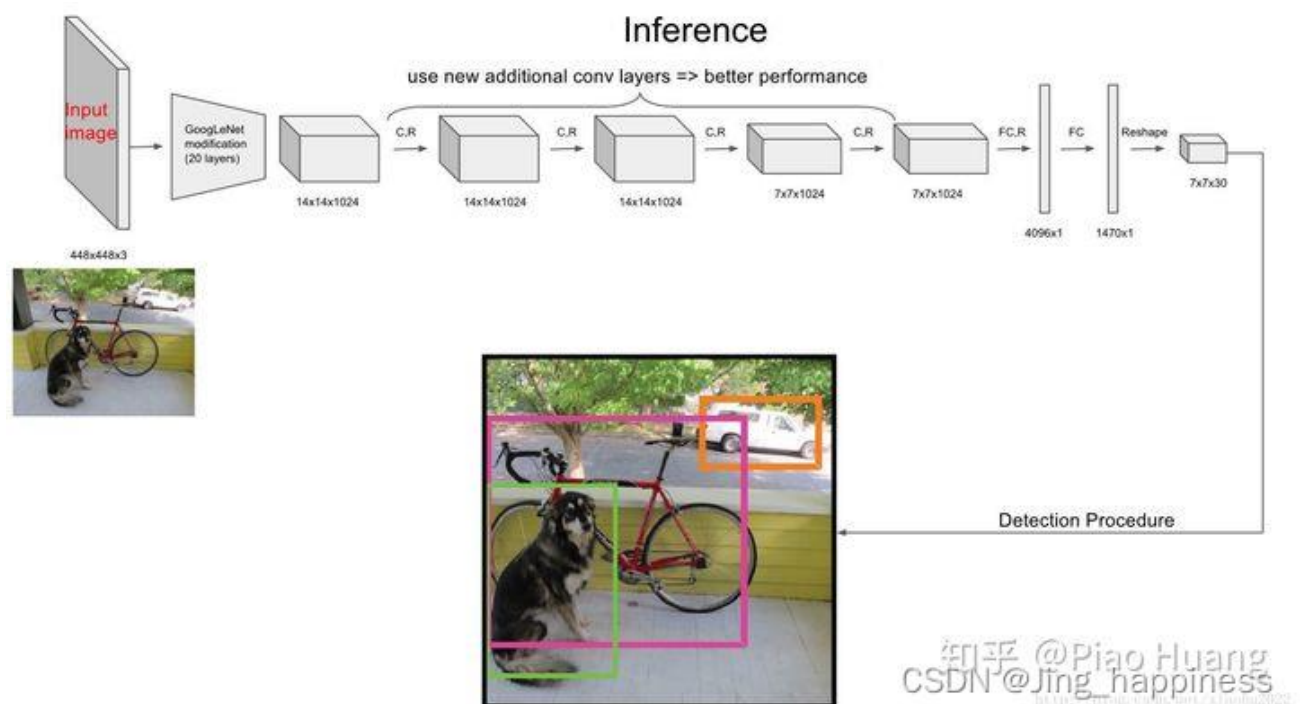


CSDN @Jing_happiness

YOLO 系列

与 R-CNN 系列最大的区别是用一个卷积神经网络结构就可以从输入图像直接预测 bounding box 和类别概率，实现了 End2End 训练。

对于网络的输出，我们可以这样理解， $30=20+2+4 \times 2$ ，其中 20 指的是 VOC 数据集的类别数，即 20 个类别的概率，2 指的是两个目标框的置信度，然后每个通道预测 2 个目标框，所以就是两个 (x,y,w,h) 即 8 个元素。那么作者在论文中所提到的网格划分是怎么体现的呢，这里要通过最后一个卷积层的输出来看，即 $7 \times 7 \times 1024$ ，可以看到特征图的尺寸是 7×7 ，那么根据卷积网络的特点，每一层的输出特征图上的每一个像素点都会对应着输入特征图的一个区域，也就是这个像素点的感受野，那么在最后一层卷积层输出特征图上，也是如此，所以我们可以认为是将原图划分成了 7×7 的网格区域，每个网格预测 20 个类别的概率，目标框置信度以及两个目标框信息，其中每个目标的中心位置都会转换至网格区域内 YOLO-1



激活函数

: leaky relu

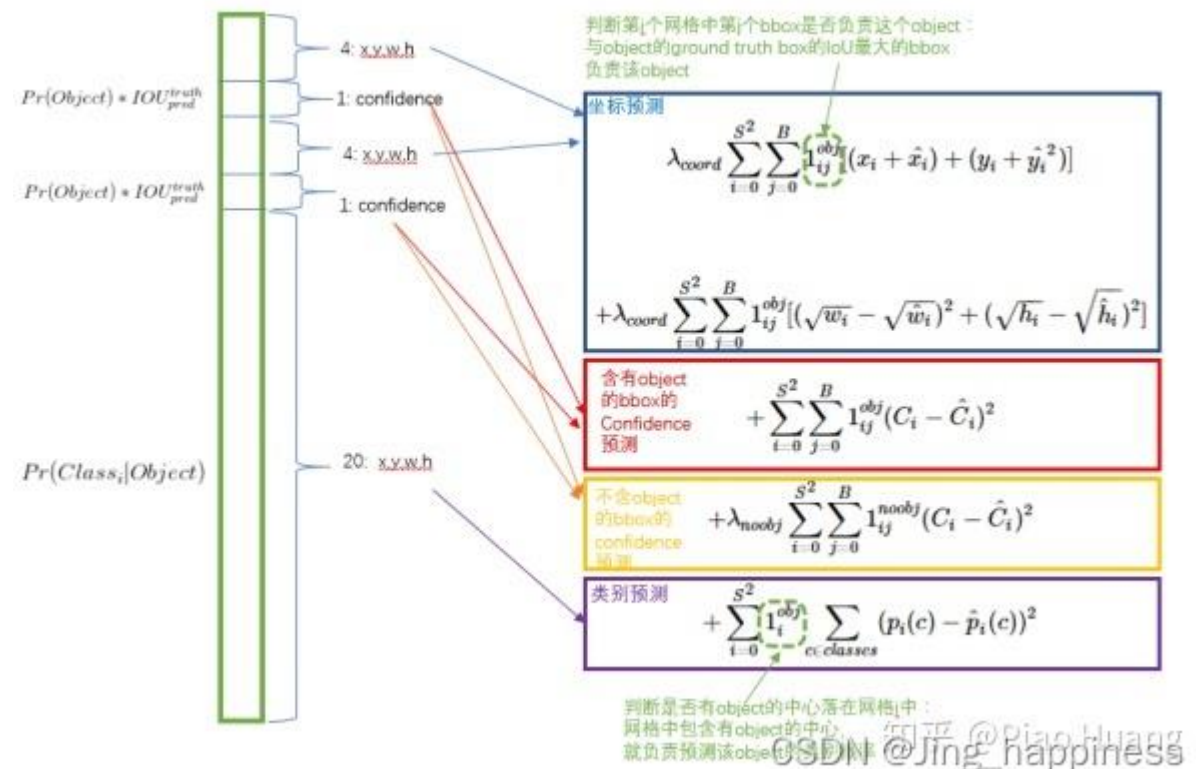
$$\text{leaky}(x) = \begin{cases} x, & x > 0 \\ 0.1x, & x \leq 0 \end{cases}$$

CSDN @Jing_happiness

知乎 @Piao Huang
CSDN @Jing_happiness

损失函数

包括：类别预测损失函数、回归的损失函数、置信度预测的损失函数



训练

在 Google net 预训练模型的基础上，

采用 dropout 策略；

lr:线性衰减（warm up 的方式，init=5e-4）

GAN

低分辨率到高分辨率图像

GP-GAN

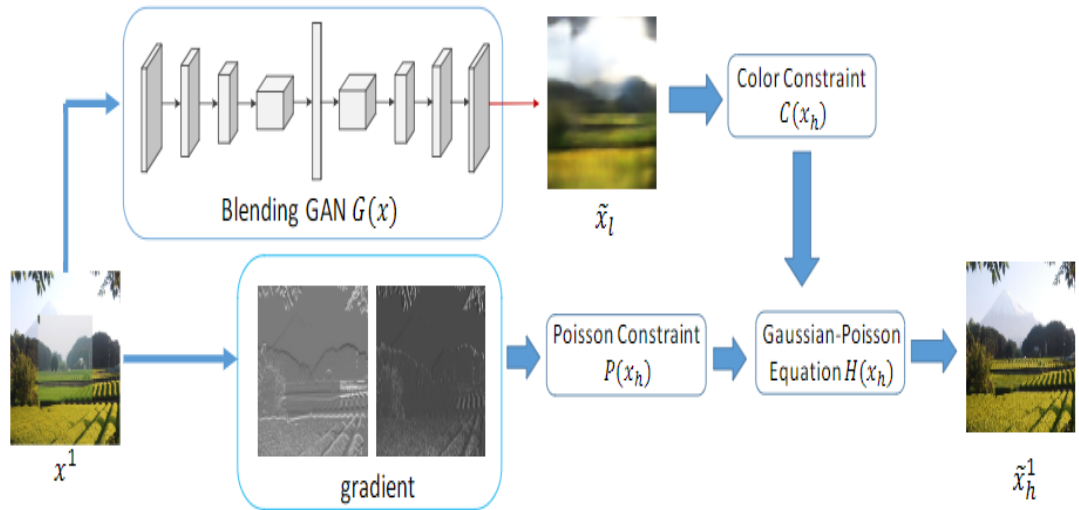


Figure 2: **Framework Overview for GP-GAN.** Given a composited image x , we first generate a low-resolution realistic image \tilde{x}_l using Blending GAN $G(x)$ with x^1 as input where x^1 is the coarsest scale in the Laplacian pyramid of x . Then we optimize the Gaussian-Poisson Equation constrained by $C(x_h)$ and $P(x_h)$ using the closed form solution to generate x_h^1 which contains many details like textures and edges. We then set \tilde{x}_l to up-sampled x_h^1 and optimize the Gaussian-Poisson Equation at a finer scale in the pyramid of x . Best viewed in color.

https://CSDN.net/@Jing_happiness/

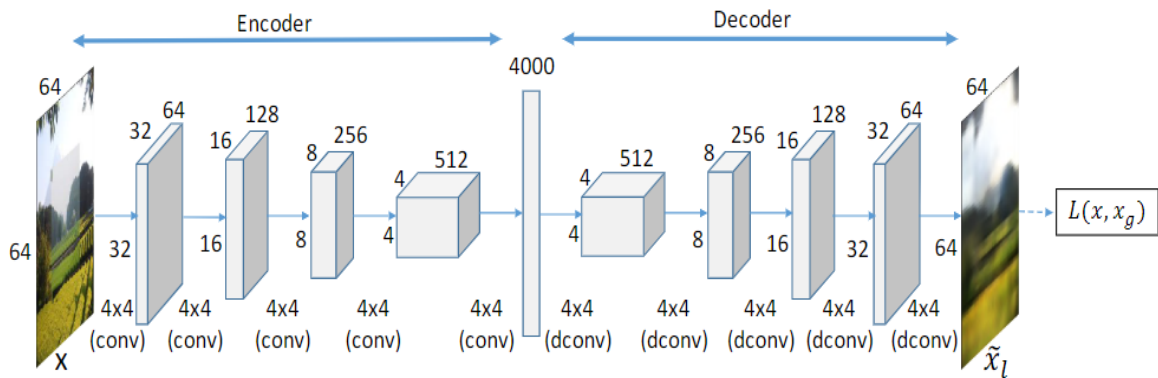


Figure 3: **Network architecture for Blending GAN $G(x)$.** We propose Blending GAN $G(x)$ by leveraging Wasserstein GAN [2] for supervised learning tasks. The encoder-decoder architecture is deployed for $G(x)$ in our experiment. Different from [24], a standard fully connected layer is inserted between the encoder and the decoder as a bottleneck to fuse the global information. The loss function $L(x, x_g)$ is defined in Equation 2, which combines the improved adversarial loss with L_2 loss.

损失函数为 L2 损失与对抗损失（Adversarial Loss）的组合。

总损失函数：

$$L(x, x_g) = \lambda L_{l_2}(x, x_g) + (1 - \lambda) L_{adv}(x, x_g)$$

L2 损失：

$$L_{l_2}(x, x_g) = \|G(x) - x_g\|_2^2$$

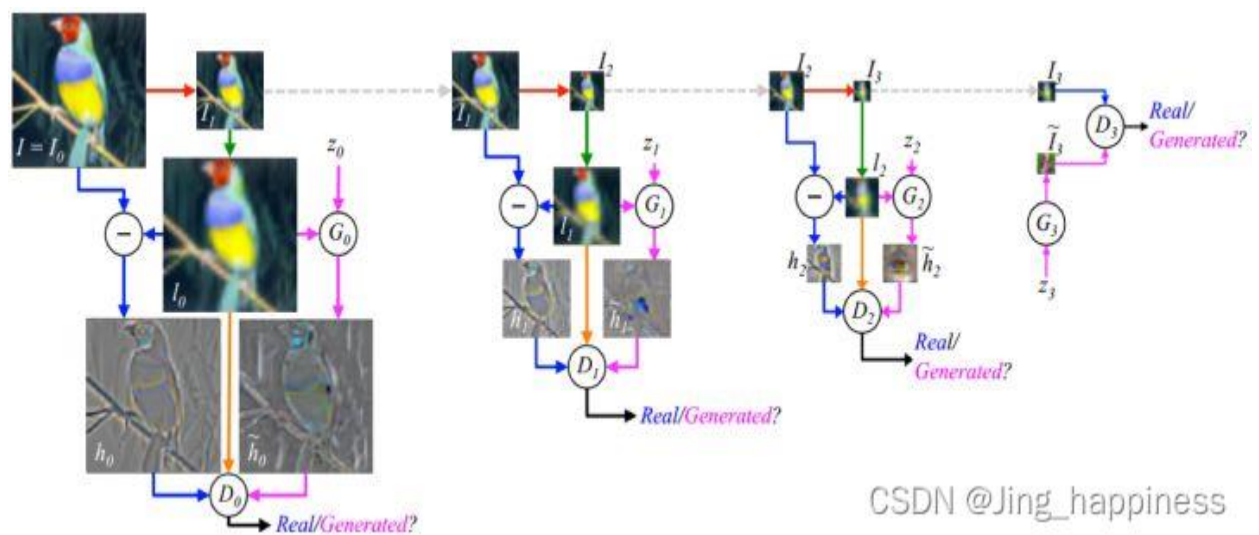
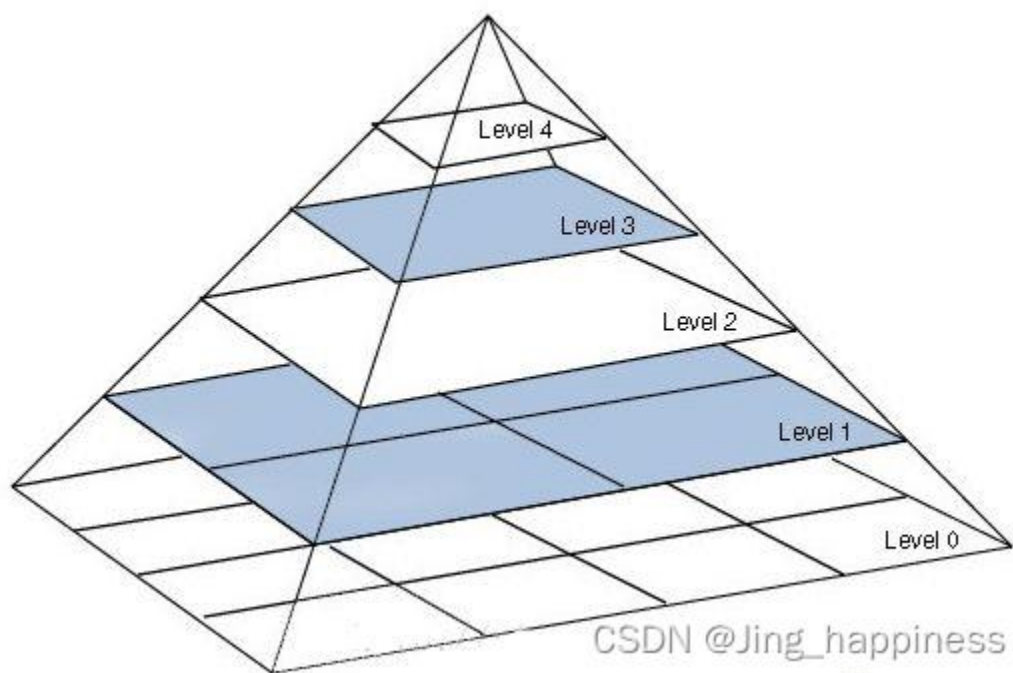
对抗损失：

$$L_{adv}(x, x_g) = \max_D E_{x \in \mathcal{X}} [D(x_g) - D(G(x))]$$

CSDN @Jing_hap

LAPGAN

一幅图像的金字塔是一系列以金字塔形状排列的分辨率逐步降低，且来源于同一张原始图的图像集合，从塔底到塔顶图像分辨率越来越低。用 $d(\cdot)$ 表示 downsample， $u(\cdot)$ 表示 upsample。



I_0 是原始输入图像（左侧图片）， G_i 表示第 i 个生成器， D_i 表示第 i 个描述器。

实验：

CIFAR-10 图片大小 3232，总共 10 类

STL 图片大小 9696，总共 10 类

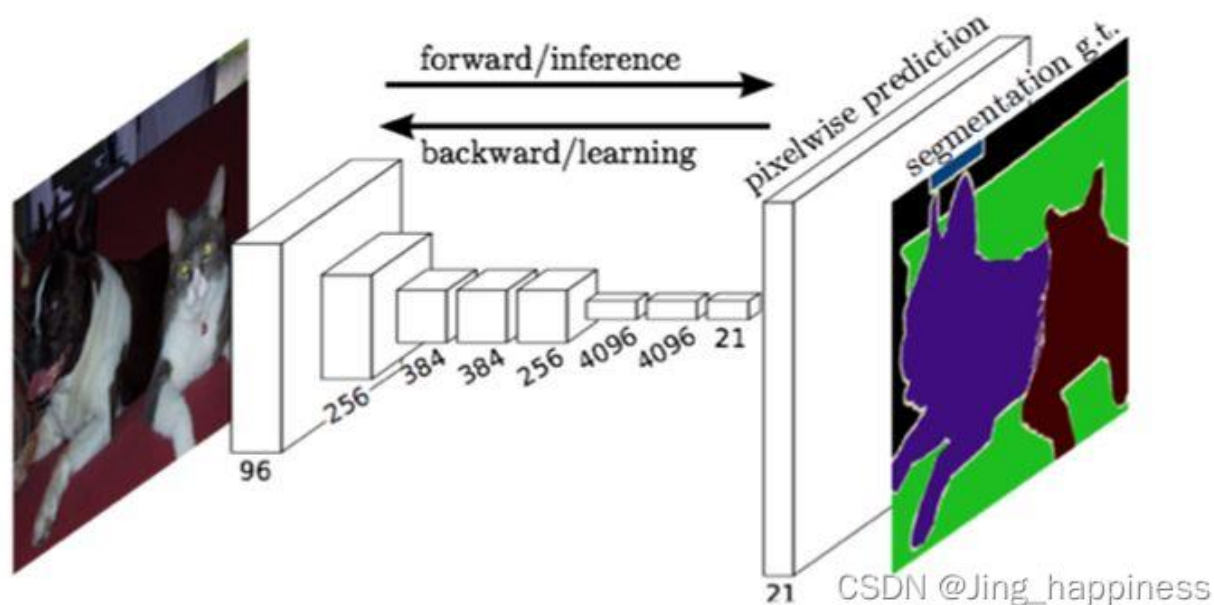
LSUN ~10M 图像，大小下采样到 64*64，10 个场景。

FCN----图像分割

FCN 对图像进行像素级的分类，从而解决了语义级别的图像分割（semantic segmentation）问题。与经典的 CNN 在卷积层之后使用全连接层得到固定长

度的特征向量进行分类（全联接层+softmax 输出）不同，FCN 可以接受任意尺寸的输入图像，采用反卷积层对最后一个卷积层的 feature map 进行上采样，使它恢复到输入图像相同的尺寸，从而可以对每个像素都产生了一个预测，同时保留了原始输入图像中的空间信息，最后在上采样的特征图上进行逐像素分类。

最后逐个像素计算 softmax 分类的损失，相当于每一个像素对应一个训练样本。下图是 Longjion 用于语义分割所采用的全卷积网络(FCN)的结构示意图：



图像分割的评分标准主要有以下四种：

Pixel accuracy (像素准确性)

IoU (Intersection over Union)

Mean IoU

Dice score

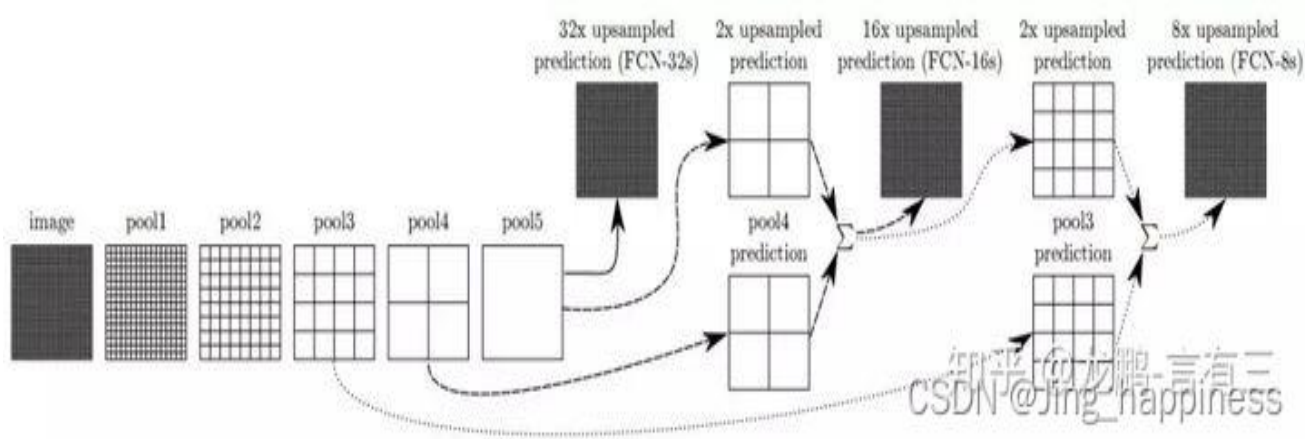
2) 连接不同尺度下的层

分类网络通常会通过设置步长的方式逐渐减小每层的空间尺寸，这种方式可以同时实现计算量的缩小和信息的浓缩。尽管这种操作对于分类任务是很有效的，但是对于分割这样需要稠密估计的任务而言，这种浓缩就未必是好事了。

比如下面这张图就是全局步长 32 下的分割效果。虽然实现了分割，但是结果很粗糙，看不出来目标的细节。



于是，为了解决这个问题，FCN 将不同全局步长下的层之间进行连接。具体网络结构如下图所示。



尽管 FCN 意义重大，在当时来讲效果也相当惊人，但是 FCN 本身仍然有许多局限。比如：

没有考虑全局信息

无法解决实例分割问题

速度远不能达到实时

不能够应对诸如 3D 点云等不定型数据

图像质量评测指标

PSNR

通常用来评价一幅图像压缩后和原图像相比质量的好坏，当然，压缩后图像一定会比原图像质量差的，所以就用这样一个评价指标来规定标准了。**PSNR 越高，压缩后失真越小。**这里主要定义了两个值，一个是均方差 MSE，另一个是

峰值信噪比 PSNR，公式如下：

PSNR 越高，图像和原图越接近。

PSNR（峰值信噪比）：用得最多，但是其值不能很好地反映人眼主观感受。

一般取值范围：20-40.值越大，视频质量越好

$$MSE = \frac{1}{mn} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} ||I(i,j) - K(i,j)||^2$$
$$PSNR = 10 \cdot \log_{10} \left(\frac{MAX_I^2}{MSE} \right) = 20 \cdot \log_{10} \left(\frac{MAX_I}{\sqrt{MSE}} \right)$$

这里的MAX通常是图像的灰度级，一般就是255了。

CSDN @Jing_happiness

SSIM（结构相似性）：

计算稍复杂，其值可以较好地反映人眼主观感受。一般取值范围：0-1.值越大，视频质量越好。经常用到图像处理中，特别在图像去噪处理中在图像相似度评价上全面超越 SNR（signal to noise ratio）和 PSNR（peak signal to noise ratio）

图像质量评价指标

update 2018-07-07 16:50:16

均方误差(MSE)和均方根误差(RMSE)和平均绝对误差(MAE)

MSE: Mean Squared Error

均方误差是指参数估计值与参数真值之差平方的期望值;

MSE 可以评价数据的变化程度，MSE 的值越小，说明预测模型描述实验数据具有更好的精确度。

RMSE

均方误差:均方根误差是均方误差的算术平方根

MAE :Mean Absolute Error

平均绝对误差是绝对误差的平均值

平均绝对误差能更好地反映预测值误差的实际情况.

SD :standard Deviation

标准差:标准差是方差的算术平方根。标准差能反映一个数据集的离散程度。平均数相同的两组组数据，标准差未必相同。

姿势估计评测指标

oks

是目前常用的人体骨骼关键点检测算法的评价指标，这个指标启发于目标检测中的 IoU 指标，目的就是为了计算真值和预测人体关键点的相似度。

OKS:

$$OKS_p = \frac{\sum_i \exp\{-d_{pi}^2 / 2S_p^2 \sigma_i^2\} \delta(v_{pi} > 0)}{\sum_i \delta(v_{pi} > 0)}$$

参数详细解释，其中：

p 表示当前图片所有groundtruth行人中id为p的人， $p \in (0, M)$ ， M 表示当前图中共有行人的数量

i 表示id为i的keypoint

d_{pi} 表示当前检测的一组关键点中id为i的关键点与groundtruth行人中id为p的人的关键点中id为i的关键点的欧式距离， $d_{pi} = \sqrt{(x'_i - x_{pi})(y'_i - y_{pi})}$ ， (x'_i, y'_i) 为当前的关键点检测结果， (x_{pi}, y_{pi}) 为groundtruth

S_p 表示groundtruth行人中id为p的人的尺度因子，其值为行人检测框面积的平方根： $S_p = \sqrt{wh}$ ， w 、 h 为检测框的宽和高

σ_i 表示id为i类型的关键点归一化因子，这个因子是通过对所有的样本集中的groundtruth关键点由人工标注与真实值存在的标准差， σ 越大表示此类型的关键点越难标注。根据[1]中所述，对coco数据集中的5000个样本统计出17类关键点的归一化因子， σ 的取值可以为：{鼻子：0.026，眼睛：0.025，耳朵：0.035，肩膀：0.079，手肘：0.072，手腕：0.062，臀部：0.107，膝盖：0.087，脚踝：0.089}，因此此值可以当作常数看待，但是使用的类型仅限这个里面。如果使用的关键点类型不在此当中，可以使用另外一种统计方法计算此值，详见下文

v_{pi} 表示groundtruth中id为p的行人第i个关键点的可见性，其中 $v_{pi} = 0$ 表示关键点未标记，可能的原因是图片中不存在，或者不确定在哪， $v_{pi} = 1$ 表示关键点无遮挡并且已经标注， $v_{pi} = 2$ 表示关键点有遮挡但已标注。同样，预测的关键点有两个属性： $v'_{pi} = 0$ 表示未预测出， $v'_{pi} = 1$ 表示预测出

$\delta(*)$ 表示如果条件*成立，那么 $\delta(*) = 1$ ，否则 $\delta(*) = 0$ ，在此处的含义是：仅计算groundtruth中已标注的关键点OKS

csdn @ling_happiness

AP(Average Precision)平均准确率

此指标用于计算测试集的精度百分比，**单人姿态估计**和**多人姿态估计**的计算方式不同。

单人姿态估计AP:

单人姿态估计，一次仅对一个行人进行估计，即在**oks**指标中 $M = 1$ ，因此一张图片中groundtruth为一个行人(GT)，对此行人进行关键点检测后会获得一组关键点(DT)，最后会计算出GT与DT的相似度**oks**为一个标量，然后人为的给定一个阈值**T**，然后可以通过所有图片的**oks**计算AP:

$$AP = \frac{\sum_p \delta(oks_p > T)}{\sum_p 1}$$

多人姿态估计AP:

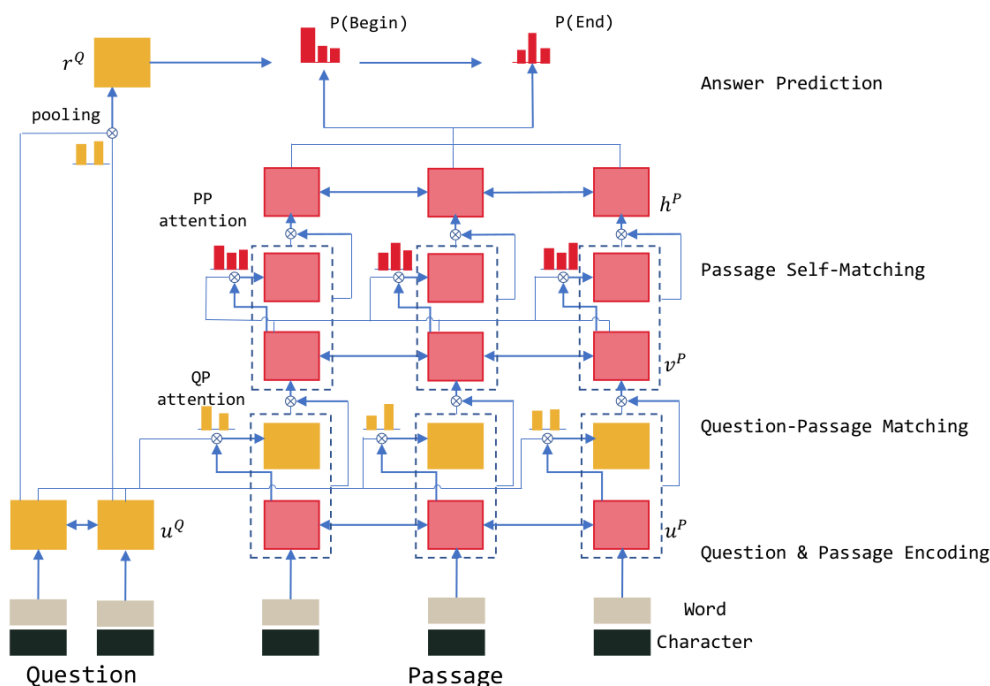
多人姿态估计，如果采用的检测方法是**自顶向下**，先把所有的人找出来再检测关键点，那么其AP计算方法如同**单人姿态估计AP**；如果采用的检测方法是**自底向上**，先把所有的关键点找出来然后再组成人，那么假设一张图片中共有**M**个人，预测出**N**个人，由于不知道预测出的**N**个人与groundtruth中的**M**个人的——对应关系，因此需要计算groundtruth中每一个人与预测的**N**个人的**oks**，那么可以获得一个大小为 $M \times N$ 的矩阵，矩阵的每一行为groundtruth中的一个人与预测结果的**N**个人的**oks**，然后找出每一行中**oks**最大的值作为当前**GT**的**oks**。最后每一个**GT**行人都有一个标量**oks**，然后人为的给定一个阈值**T**，然后可以通过所有图片中的所有行人计算AP:

$$AP = \frac{\sum_m \sum_p \delta(oks_p > T)}{\sum_m \sum_p 1}$$

CSDN @Jing_happiness

阅读理解模型

R-Net



作者引入了一个端到端的网络模型来解决基于给定文章回答问题的阅读理解问答任务。首先使用门控的基于注意力的循环网络来匹配问题和文章，以获得问题感知的文章表示。

作者提出了一个自匹配的注意力机制来改善文章表示，通过与文章自己进行匹配，这能有效的编码整个文章的信息。
最后应用指针网络来从文章中定位答案。
模型由四部分组成：

- RNN 编码器分别为问题和文章编码(Question & Passage Encoding)
- 门控匹配层进行问题和文章的匹配(Question-Passage Matching)
- 自匹配层为整篇文章进行信息压缩(Passage Self-Matching)
- 基于指针网络的答案边界预测层(Answer Prediction)

FusionNet

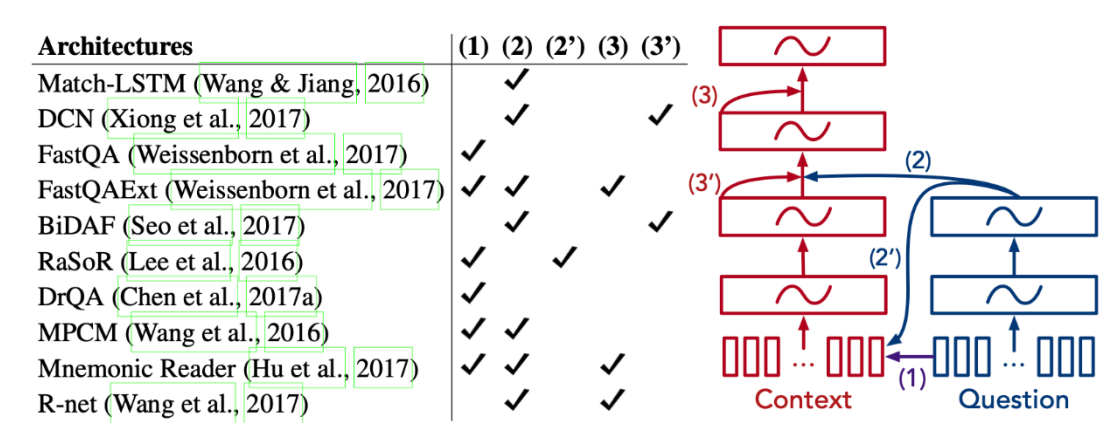


Table 1: A summarized view on the fusion processes used in several state-of-the-art architectures.

Figure 2: A conceptual architecture illustrating recent advances in MRC.

图 2 显示了最先进架构的概念架构，该架构由三个组件构成。

- 输入向量：上下文(context)和问题中每个单词的嵌入向量。
- 集成组件： 图中的矩形框，通常是基于 RNN 实现的。
- 融合过程： 带有数字的箭头(1),(2),(2'),(3),(3')。从箭头起始位置向箭头指向位置融合

现在主要的三种融合过程。上图左边的表显示了不同网络结构使用的融合过程。

(1)单词级融合 通过向上下文(Context,或者说文章)提供直接的单词信息，我们可以快速放大上下文中更相关的区域。然而，如果一个单词基于上下文有不同的语义，此时可能用处不大。许多单词级别的融合不是基于注意力，例如有人在上下文单词中添加二进制特征，指示每个上下文单词是否出现在问题中。

(2)高层融合 融合问题中语义信息的上下文可以帮助我们找到正确的答案。但高层信息没有单词信息准确，这可能会导致模型对细节的了解不够。

(2')**高层融合(可选)** 类似地，我们也能将问题的高层信息融入上下文的单词信息中。

(3)**自提升融合** 由于上下文可能会很长，而较远部分的文本可能需要互相依赖来充分理解文本信息。**最近的研究建议将上下文融合到自身**。由于上下文包含过多的信息，一个常见的选择是在融合问题 Q 后执行自提升融合(self-boosted fusion)。这使我们能够更好地了解与问题相关的区域。

(3')**自提升融合(可选)** 另一种选择是直接在问题 Q 上调整自提升融合过程，比如 DCN4 论文中提出的共同注意力机制(coattention mechanism)。然后，我们可以在融合问题信息之前执行自提升融合

全感知融合网络

文本 A 是阅读理解中的上下文 C，文本 B 是问题 Q，每个圆圈是对左图中模块的详细说明。

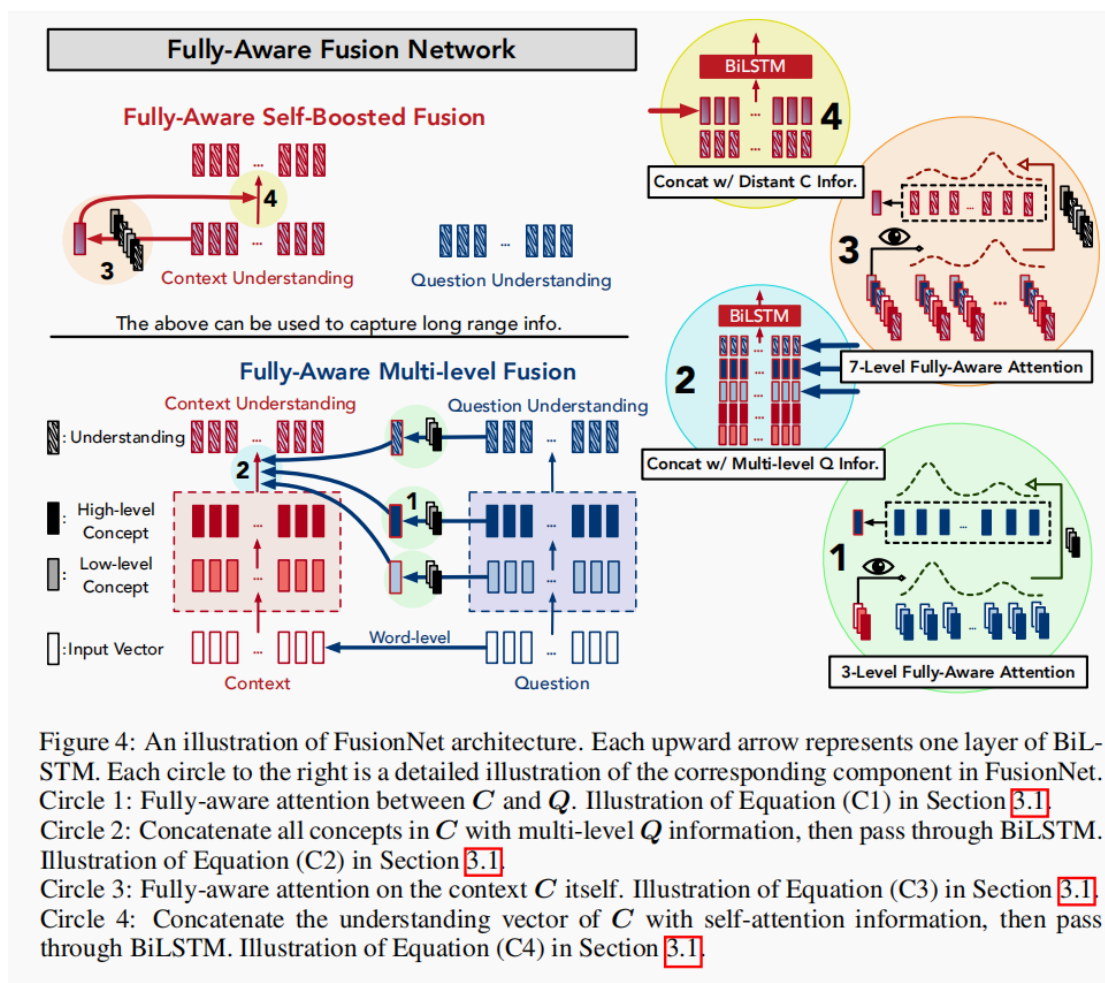
FusionNet 图示。每个向上的箭头代表 BiLSTM 的一层。右边的每个圆圈都是 FusionNet 中相应组件的详细说明。

圆圈 1： C 和 Q 之间的全感知注意力。描述了下面的公式(C1)

圆圈 2： 将 C 中的所有信息与多层次的 Q 信息连接起来，然后经过 BiLSTM。描述了下面的公式(C2)

圆圈 3： 上下文 C 的自全感知注意。描述了下面的公式(C3)

圆圈 4： 将 C 的理解向量和自注意信息拼接起来，然后经过 BiLSTM。描述了下面的公式(C4)



输入向量 首先，每个 C 和 Q 中的单词被转换为一个输入向量 w 。

全感知多层融合：单词级 在多层融合时，作者分别考虑了单词级融合和高层融合的情况。单词级融合告诉 C 关于 Q 中有什么类型的单词。

阅读 在阅读组件中，作者分别为 C 和 Q 使用一个 BiLSTM 来形成低层和高层信息。

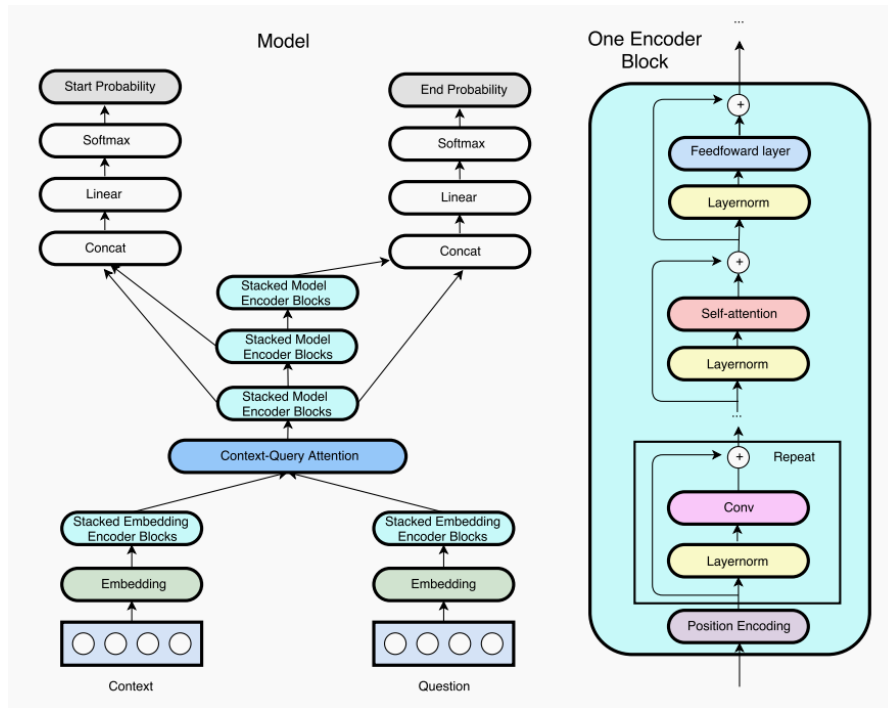
问题理解

全感知多层融合：高层 这个组件通过对单词历史的全感知注意力，将问题 Q 中的所有高层信息与上下文 C 融合。

全感知自提升融合 我们现在使用自提升(self-boosted)融合来考虑上下文中的远距离部分。

QA-Net

在此论文之前端到端的机器阅读和问答模型(像之前介绍的 BiDAF)主要基于带注意力的 RNN 网络，这种架构有一个很重要的缺点是训练和推理速度慢。此论文提出了一个不含 RNN 网络的架构，其编码器仅包含卷积和自注意力(self-attention)，称为 QANet



模型架构如上图左。

首先介绍符号定义。给定一个包含 n 个单词的上下文段落 $C = \{c_1, c_2, \dots, c_n\}$ 和包含 m 个单词的查询语句 $Q = \{q_1, q_2, \dots, q_m\}$ ，以及从原始段落 C 中抽取的输出片段(span) $S = \{c_i, c_{i+1}, \dots, c_{i+j}\}$ 。

模型包含五个组件：嵌入层、嵌入编码器层、上下文-查询注意力层、模型编码器层和输出层。

输入嵌入层

作者通过拼接字符嵌入(character embedding)和词嵌入得到每个单词的嵌入向量。词嵌入用的是预训练好的 GloVe 向量，并且不会随着训练进行更新。而字符嵌入是可学习的，

嵌入编码器层

编码器层由以下构建块堆叠而成：[卷积层 $\times \#$ + 自注意层 + 前馈层]，如图 1 右所示。卷积层 $\times \#$ 表示重复堆叠了 $\#$ 次。卷积层基于深度可分离卷积；自注意层基于多头注意力机制。每个基础操作层(卷积/自注意/前馈)周围都有残差连接。同时应用了层归一化(layer norm)，对于每个操作 f ，输出是 $f(\text{layernorm}(x)) + x$ 。

上下文-查询注意力层

该模块几乎是阅读理解模型的标配。使用 \mathbf{C} 和 \mathbf{Q} 来表示编码的上下文和查询。

该模块几乎是阅读理解模型的标配。使用 \mathbf{C} 和 \mathbf{Q} 来表示编码的上下文和查询。上下文到查询之间的注意力基于如下方式构建：首先计算上下文和查询单词对之间的相似度，得到一个相似矩阵 $\mathbf{S} \in \mathbb{R}^{n \times m}$ 。然后应用softmax函数对 \mathbf{S} 的每行进行归一化，得到新的矩阵 $\bar{\mathbf{S}}$ 。然后上下文-查询注意力计算为 $\mathbf{A} = \bar{\mathbf{S}} \cdot \mathbf{Q}^T \in \mathbb{R}^{n \times d}$ 。

相似度函数为：

$$f(q, c) = W_0[q, c, q \odot c] \quad (21)$$

接着像DCN模型还计算了查询到上下文的注意力。具体做法是对 \mathbf{S} 的列进行归一化，得到矩阵 $\tilde{\mathbf{S}}$ ，然后查询到上下文的注意力矩阵 $\mathbf{B} = \bar{\mathbf{S}} \cdot \tilde{\mathbf{S}}^T \cdot \mathbf{C}^T$ 。

模型编码器层

输出层

该层是任务相关的。在解决SQuAD任务中，作者采用了Bi-DAF的策略来预测上下文每个位置作为答案片段开始和结束的概率。具体做法为：

$$p^1 = \text{softmax}(W_1[M_0; M_1]), \quad p^2 = \text{softmax}(W_2[M_0; M_2]) \quad (22)$$

其实 W_1 和 W_2 是可训练的权重， M_0, M_1, M_2 分别是前面三个模型编码器的输出，由下到上。

然后用这两个概率相乘得到片段得分。

最后，定义训练损失为预测分布对应的真实开始和结束位置的负对数似然的之和，并求所有样本上的均值：

$$L(\theta) = -\frac{1}{N} \sum_i^N [\log(p_{y_i^1}^1) + \log(p_{y_i^2}^2)] \quad (23)$$

通常的阅读理解任务包含了四个模块。

分别是：

1) Embeddings: 该模块以上下文和问题为输入，通过各种方法输出上下文和问题嵌入。经典的词表示方法如 **one-hot** 和 **Word2Vec**，有时结合其他语言特征，即词性、名称实体和问题类别，通常用于表示词中的语义和句法信息。此外，由大型语料库预训练的上下文词表示在编码上下文信息方面也表现出良好的性能。

2) 特征提取: 在嵌入模块之后，上下文和问题嵌入被馈送到特征提取模块。为了更好地理解上下文和问题，该模块旨在提取更多上下文信息。一些典型的深度神经网络，例如循环神经网络 (**RNN**) 和卷积神经网络 (**CNN**)，被应用于从上下文和问题嵌入中进一步挖掘上下文特征。

3) 上下文-问题交互: 上下文和问题之间的相关性在预测答案中起着重要作用。有了这些信息，机器就可以找出上下文中的哪些部分对回答问题更重要。为了实现该目标，该模块中广泛使用单向或双向注意力机制来强调与查询相关的上下文部分。为了充分提取它们的相关性，上下文和问题之间的交互有时会涉及多跳，这模拟了人类理解的重读过程。

4) 答案预测: 答案预测模块是 **MRC** 系统的最后一个组件，它根据先前模块积累的所有信息输出最终答案。由于 **MRC** 任务可以根据答案形式进行分类，因此该模块与不同的任务高度相关。对于完形填空测试，该模块的输出是原始上下文中的单词或实体，而多项选择任务需要从候选答案中选择正确答案。在跨度提取方面，该模块提取给定上下文的子序列作为答案。该模块中使用了一些生成技术来完成自由回答任务，因为它对回答形式几乎没有限制

阅读理解评估指标

- 1) 完形填空、多项选择：精度 **acc**
- 2) **span extraction**: **F1** 值
- 3) **free answer**: **ROUGE**、**BELU**（凡事可以评估两个序列之间重叠程度的都可以）

阅读理解数据集

(1) 完型填空：原文中挖出一个空来，由机器根据对文章上下文的理解去补全。

代表数据集有 **CNN/Daily Mail**

(2) 多项选择：每篇文章对应多个问题，每个问题有多个候选 答案，机器需要在这些候选答案中找到最合适的那个。代 表数据集有 **RACE**

(3) 区域预测：也称为抽取式问答（**Extractive QA**），即给定文章和问题，机器需要在文章中找到答案对应的区域（**span**），给出开始位置和结束位

置。代表数据集有 **SQuAD** (**Stanford Question Answering Dataset**)

- (4) 自由形式：不限定问题所处的段落，即一个问题可能是需要理解多个段落甚至多篇文章。代表数据集有 **DuReader** (百度)，**MS MARCO** (微软)

CNN 中的变形

反卷积：在 deep learning 上，主要用于以下三个方面。unsupervised learning：重构图

像；CNN 可视化：将 conv 中得到的 feature map 还原到像素空间，来观察特定的 feature

map 对哪些 pattern 的图片敏感；Upsampling：上采样。

在前向传播是使用 **C**，后向传播时使用 **C^T** 便是普通的 **conv**。反之，则是 **deconv**。

- ◆ $CX = Y, X = C^T Y$

- ◆ So C and $C^T \Rightarrow \text{conv}$, C^T and $(C^T)^T \Rightarrow \text{deconv}$.

反卷积又被成为分数步长卷积，它实现的是调转一个卷积的前向传递和反向传递过程。一种表述方式是——卷积核定义了一个卷积过程，置于它普通卷积还是反卷积取决于其正向、反向传递过程时如何计算的。举例来说，尽管一个卷积核定义了一种卷积运算，其前向传递和反向传递过程分别通过乘以 **C** 和 **C^T** 得到，但它仍可以定义另一个卷积运算，其前向传递和反向传递过程分别通过乘以 **C^T** 和 **(C^T)^T** 的到。

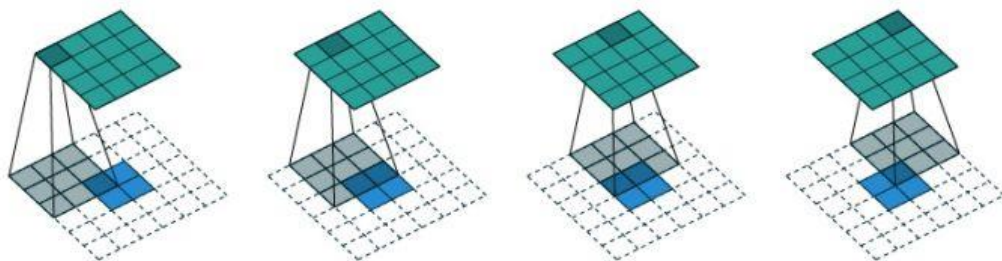


Figure 4.1: The transpose of convolving a 3×3 kernel over a 4×4 input using unit strides (i.e., $i = 4$, $k = 3$, $s = 1$ and $p = 0$). It is equivalent to convolving a 3×3 kernel over a 2×2 input padded with a 2×2 border of zeros using unit strides (i.e., $i' = 2$, $k' = k$, $s' = 1$ and $p' = 2$).

空洞卷积 (Dilated convolutions)

空洞卷积通过在卷积核的元素间插入空间对卷积核进行“膨胀”，这个“膨胀率”通过控制一个超参数 d 实现。空洞卷积的实现方式有多种，但是一般插入的空间是 $d = 1$ ，当 $d = 1$ 是，它就是一个普通卷积。空洞卷积是一种可以轻易扩大卷积层输出感受野的方式，其比较有效的用法是一层层堆叠的多个空洞卷积层的组合。

为了理解空洞卷积的计算方式，了解参数 d 对扩充后卷积核大小的影响是比较合适的途径。一个卷积核 k 经过参数 d 进行扩充之后得到的实际核大小为: $k' = k + (k - 1)(d - 1)$ 然后，我们可以利用式6，计算其输出大小，如式15:

$$o = \left\lceil \frac{i + 2p - k - (k - 1)(d - 1)}{s} \right\rceil + 1.$$

图5.1 给出了一个 $i = 7$, $k = 3$ 和 $d = 2$ 的例子。

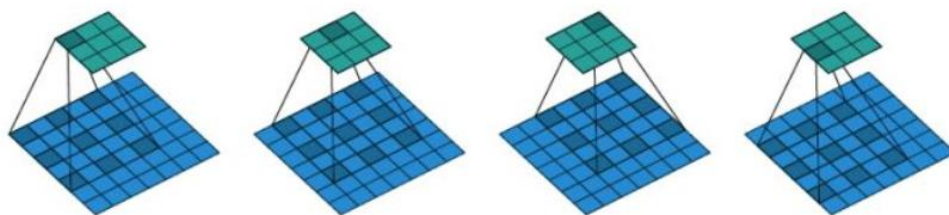
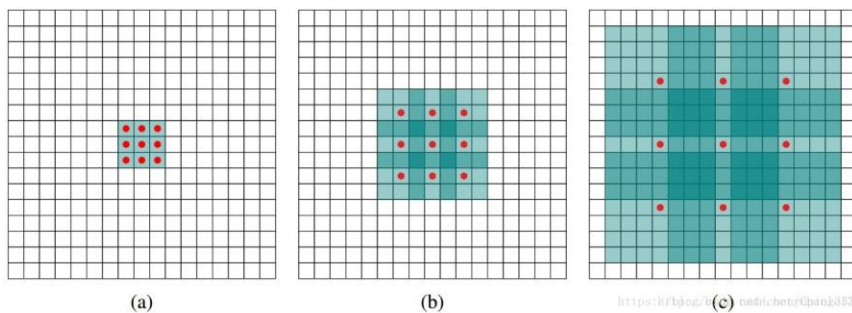


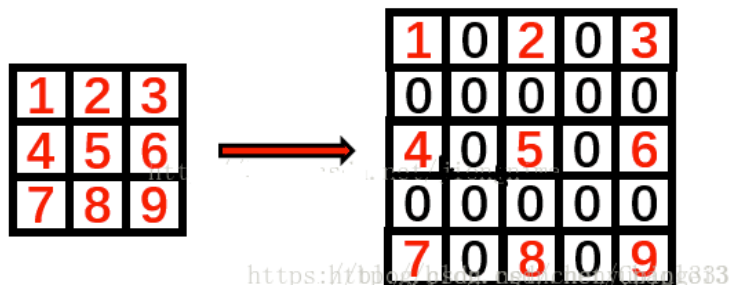
Figure 5.1: (Dilated convolution) Convolving a 3×3 kernel over a 7×7 input with a dilation factor of 2 (i.e., $i = 7$, $k = 3$, $d = 2$, $s = 1$ and $p = 0$).



(a) 图对应 3x3 的 1-dilated conv，和普通的卷积操作一样。(b)图对应 3x3 的 2-dilated conv，实际的卷积 kernel size 还是 3x3，但是空洞为 1，需要注意的是空洞的位置全填进去 0，填入 0 之后再卷积即可。【此变化见下图】

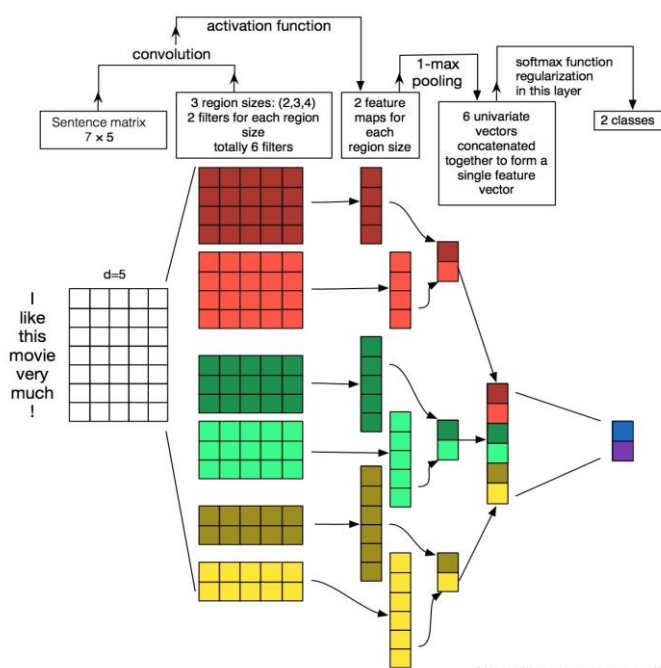
(c) 图是 4-dilated conv 操作。

(b) dilated=2 时具体的操作，即按照下图在空洞位置填入 0 之后，然后直接卷积就可以了



不同 kernel 时的卷积操作：

对于 $k=4$ ，见图中红色的大矩阵，卷积核大小为 4×5 ，步长为 1。这里是针对输入从上到下扫一遍，输出的向量大小为 $((7-4)/1+1) \times 1 = 4 \times 1$ ，最后经过一个卷积核大小为 4 的 max_pooling，变成 1 个值。最后获得 6 个值，进行拼接，在经过一个全连接层，输出 2 个类别的概率。



Pool 的变形

最大池化:

平均池化:

随机池化: 只需对 **feature map** 中的元素按照其概率值大小随机选择, 即元素值大的被选中的概率也大

SoftPool: 权重与相应的激活值一起用作非线性变换。较高的激活比较低的激活占更多的主导地位

步骤:

1. 计算权重 w_i

$$w_i = \frac{e^{a_i}}{\sum_{j \in R} e^{a_j}}$$

2. 为所有邻域内激活值加权求和

$$\tilde{a} = \sum_{i \in R} w_i * a_i$$

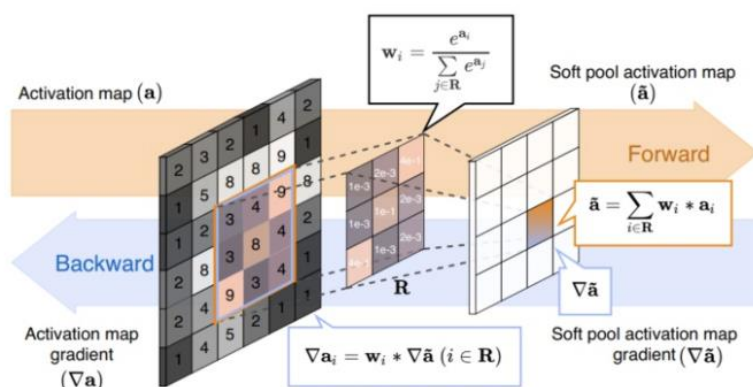


Figure 4. **SoftPool operation overview**. In forward operation, in **orange**, the kernel uses the exponential softmax value of each activation as weight and then calculates the weighted sum for the region **R**. The same weights are also used for the gradients ($\nabla \tilde{a}_i$), denoted in **blue**. The activation gradients are proportional to the calculated softmax weights.

重叠池化: 相对于传统的 no-overlapping pooling, 采用 Overlapping Pooling 不仅可以提升预测精度, 同时一定程度上可以减缓过拟合。相比于正常池化 (步长 $s=2$, 窗口 $z=2$) 重叠池化(步长 $s=2$, 窗口 $z=3$) 可以减少 top-1, top-5 分别为 0.4% 和 0.3%; 重叠池化可以在一定程度避免过拟合

金字塔池化 Spatial Pyramid Pooling: 在一般的 CNN 结构中，在卷积层后面通常连接着全连接。而全连接层的特征数是固定的，所以在网络输入的时候，会固定输入的大小(fixed-size)。而 SPP 利用多尺度 pooling 然后 reshape 并拼接，得到固定大小的特征向量（256 应该表示的是通道数）

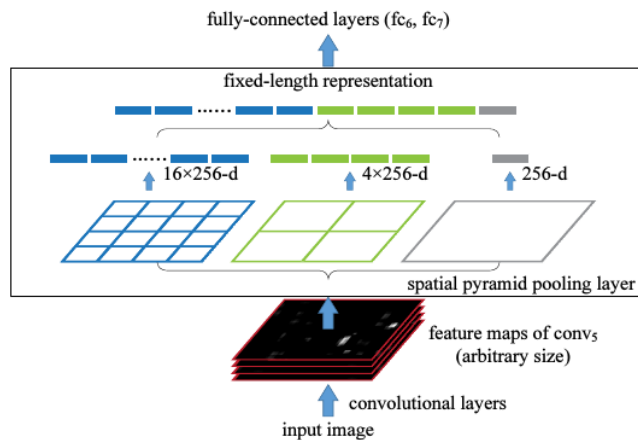


Figure 3: A network structure with a **spatial pyramid pooling layer**. Here 256 is the filter number of the conv₅ layer, and conv₅ is the last convolutional layer.

对角线那种的 pool: 这种从左向右，从上到下的 pool 方式，这符合了 language 的习惯，我觉得，是可以借鉴的，无奈水平有限，没有实现。

CornerNet



Fig. 2 Often there is no local evidence to determine the location of a bounding box corner. We address this issue by proposing a new type of pooling layer.

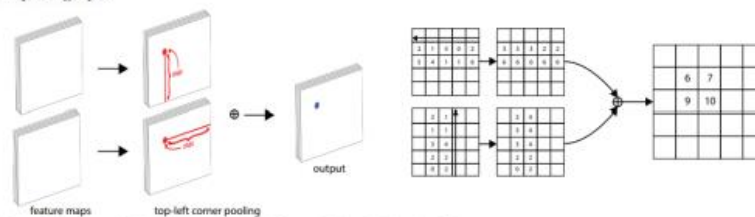


Fig. 3 Corner pooling: for each channel, we take the maximum values (red dots) in two directions (red lines), each from a separate feature map, and add the two maximums together (blue dot).

Hei Law, Jia Deng. CornerNet: Detecting Objects as Paired Keypoints. ECCV (14) 2018: 765-781

适应性焦点损失函数：（TH 是额外设置的一个类别）

$$\mathcal{L}_{RE} = \sum_{r_i \in \mathcal{P}_T} (1 - P(r_i))^\gamma \log(P(r_i)) + \log(P(r_{TH})) \quad (11)$$

where γ is a hyper-parameter. Our loss is designed to focus more on the low-confidence classes. If $P(r_i)$ is low, the loss contribution from the relevant class will be higher, which enables a better optimization for long-tail classes.

多分类损失函数：

这时候我们之前研究的《将“softmax+交叉熵”推广到多标签分类问题》就可以派上用场了。简单来说，这是一个用于多标签分类的损失函数，它是单目标多分类交叉熵的推广，特别适合总类别数很大、目标类别数较小的多标签分类问题。其形式也不复杂，在GlobalPointer的场景，它为

$$\log \left(1 + \sum_{(i,j) \in P_\alpha} e^{-s_\alpha(i,j)} \right) + \log \left(1 + \sum_{(i,j) \in Q_\alpha} e^{s_\alpha(i,j)} \right) \quad (3)$$

其中 P_α 是该样本的所有类型为 α 的实体的首尾集合， Q_α 是该样本的所有非实体或者类型非 α 的实体的首尾集合，注意我们只需要考虑 $i \leq j$ 的组合，即

$$\begin{aligned} \Omega &= \{(i, j) \mid 1 \leq i \leq j \leq n\} \\ P_\alpha &= \{(i, j) \mid t_{[i:j]} \text{ 是类型为 } \alpha \text{ 的实体}\} \\ Q_\alpha &= \Omega - P_\alpha \end{aligned} \quad (4)$$