Vision Transformer

1.ViT:An image is worth 16x16 words: Transformers for image recognition at scale(2020)

论文链接

本文提出了Vision Transformer，将transformer架构应用到图片分类问题，除了预处理不同，其余就是一个用于分类的transformer编码器

- 想感慨的是，这篇论文是除了Yolo v3那个技术报告外读的最顺畅的一个。一方面，ViT尽量不改变transformer结构（为了方便的直接使用nlp领域已经在硬件上高效实现的transformer结构），另一方面attention is all you need是我读的第一篇论文，读的很仔细，还印象深刻了属于是。
- 预处理：为了得到序列输入，将一张图片分割为多个patch，维度为**patch数量*(patch长*宽*通道数**，将一个patch的特征作为一个token，且通过可训练的线性映射得到D维patch embedding；为了保留位置信

息，ViT也使用了1维position embedding（2维效果没啥提升）；为了实现分类任务，在序列开始加入了一个可训练的[class]token，其最终状态作为分类的特征

- inductive bias:文中认为，CNN具有translation equivariance和locality等inductive bias（这是模型自身的一种先验），这是优点但也会受限（不如模型自己学习到）。transformer的优势在于inductive bias更少（只有MLP和position embedding），空间关系必须从头开始学，因此在大数据集上训练时优于CNN（更好的先验）。
- 微调：在微调时，remove预训练的分类头然后重新初始化进行训练。当训练的图像分辨率高于预训练时，为了保证预训练的position embedding有效，在保持patch-size不变的同时，根据patch的相对位置对embedding进行二维插值
- 论文中提到，当在中等数据集上训练时，transformer的表现不如CNN，但优势体现在数据集更大的时候。ViT通过在大型数据集上预训练，后微调得到了sota表现。
- 本文还提到一种混合模型，先用CNN提取patch的特征，再对其patch & position embedding作为输入

2.Swin Transformer: Hierarchical Vision Transformer using Shifted Windows(2021)

论文链接

本文提出了一种新的vision transformer结构Swin transformer，利用shifted window降低计算复杂度，并通过patch merge获得多尺度特征图，后续可以类似于FPN或U-Net方法进行dense prediction任务

- 背景：本文认为，将transformer应用于Vision时需要考虑两个域之间的两个差别，一为视觉实体具有不同的尺寸，二为视觉任务多需要高分辨率输入，而transformer对输入为平方复杂度。为了解决这两个问题，Swin transformer分别使用了层次特征图和计算局部self-attention的方法
- 结构：预处理与ViT类似，将图片分为patch后计算embedding（在这里无需加入position embedding），输入两个级联的Swin transformer block后进行patch merge，即令相邻的patch(2*2=4个)concatenation成4d张量后经线性层降为2d，从而使特征图长宽变为一半，相当于步长为2的下采样，将结果再输入两个级联的Swin transformer block，重复这个过程
- Swin transformer block包括级联的两部分，他们的多头自注意力层(MSA)不同。首先将输入第一个transformer block，其MSA为w-MSA，对每个无重叠的window(每个window包含M*M个patch)分别计算自注意力；将第一个block的结果输入第二个，其MSA为SW-MSA，对特征图进行shifted window分割，后对新的window（许多window尺寸小于M*M，具体看论文）计算自注意力
- w-MSA使自注意力的计算转为线性复杂度，SW-MSA建立w-MSA的不同window之间的关系，丰富了全局特征。
- 论文中提出了一种高效mask方法计算shifted window的自注意力，具体看论文
- 本文使用了相对位置偏差，在计算自注意力时加入。因为M^2与M^2的patch之间有(2M-1)*(2M-1)种相对位置关系（每个维度2M-1），所以训练一个(2M-1)*(2M-1)维度的bias矩阵，计算时从中取值即可

Object Detection

1.R-CNN:Rich feature hierarchies for accurate object detection and semantic segmentation(2014)

论文链接

本文提出R-CNN方法，将CNN方法引入目标检测领域，大大提高了目标检测效果，以此为开始引出一系列Two-Stage Object Detection工作。

- 整个训练过程可大致分为四（三）个阶段：将CNN网络在辅助大数据集上进行有监督（image-level）的预训练（或直接使用AlexNet的参数）；将原CNN网络的全连接层换为特定于目标检测任务的类别数（N+1），并输入region proposals进行微调（此时正例和负例的判定限制较松），使用softmax损失；输入region proposals并通过CNN输出的特征（N+1维）训练SVM，输出每个类别的分数，此过程中正例和负例的判定要求严格（直接使用CNN的输出softmax误差较大）；将最后一个卷积层输出的特征训练bounding-box regression
- 测试时，对每张输入图片提取2k左右region proposals；将region proposals缩放为特定大小并输入CNN网络提取特征；将提取出的特征输入SVM进行分类打分和NMS，再将剩下的proposals最后一个卷积层输出的特征进行bounding-box regression，对边界框进行微调
- 在本文提出的时代，目标检测的有标记数据少，过去往往通过无监督方法预训练，本文提出在其他数据集上进行有监督的预训练（使用辅助任务，如分类）再进行特定任务的微调更有利于训练大容量CNN（在特定任务数据稀少时）
- 本文使用了selective search方法提取region proposals，大致是先将图片过分割，再通过一些启发式的规则进行合并。通过该方法得到的region proposals尺寸不能直接作为CNN的输入，本文采取了先padding上下文再各向异缩放的方法。
- 本文使用CNN网络（AlexNet）进行特征提取，过去常用人工设定的特征，如SIFT,HOG

> SIFT和HOG未了解过，之后读原论文了解一下

- 根据设定好的交并比阈值区分正例和负例来对SVM训练，训练时每个batch的正例和负例的比例确定
- bbox回归希望对region proposals进行微调，具体细节见论文附录，主要训练了四个线性层作为平移/缩放因子

> 提到了DPM，未了解过，不过似乎和bbox回归的思想差不多，且在DL时代“过时”了，便不读原论文了

2.SPP-Net:Spatial pyramid pooling in deep convolutional networks for visual recognition(2015)

论文链接

本文将SPM方法(Spatial pyramid matching)应用到CNN中，提出了SPP-Net，在与R-CNN效果差不多的同时大大加速

- SPP(Spatial pyramid pooling)，将不同尺寸的输入划分为块（已设定好的数量），对每个块分别池化
- SPP的优势：将可变长度的输入转化为固定长度的输出，且保留空间特征；可以使用多个尺寸的图片进行训练，对目标变形更有鲁棒性；可以在多个尺度（块的数量）提取特征
- 每张图片只需用CNN提取一次特征，将原图的region proposals对应到特征图，即可获得其CNN特征，将特征图不同尺寸的region proposals通过SPP池化，将得到的不同尺寸的represention拼接起来，即可获得相同尺寸的特征向量，再送入FC分类
- SPP-Net对每张图片只提取一次特征，大大提高了效率，但训练阶段和R-CNN一样复杂，在Fast R-CNN得到解决

3.Fast R-CNN(2015)

论文链接

Fast R-CNN比R-CNN更快，更准确

- 训练过程：将一张图片和RoIs（也是通过selective search）输入CNN（将最后一个池化层换为Roi池化层），在卷积层输出的特征图上找到原图Roi对应的Roi（按CNN下采样的比例缩放），将特征图的Roi池

化后（统一尺寸）输入全连接层，分别得到softmax分类和bbox回归的偏移量，将二者的共同损失同时反向传播。(计算回归损失时使用了smooth L1 loss)

- 更快的原因：Fast R-CNN利用分层采样，对每张图片只需要前向传递CNN一次便可提取多个Rio对应的特征；使用多任务损失，同时更新整个网络的参数，无需像R-CNN一样分多个阶段
- Roi池化层，将特征图大小不同的Roi分块最大池化统一尺寸，文中亦讨论了其反向传播
- Fast R-CNN使用了VGG主干
- 推理时全连接层耗时很长，利用奇异值分解加速（分解成两个小的全连接层）

4.Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks(2015)

论文链接

基于Region的CNN在目标检测取得了很好的效果，但传统Region proposals方法（如selective search）成为推理时的时间瓶颈，本文提出RPNs(Region Proposals Networks)，在和检测网络（如Fast R-CNN）共享CNN特征的基础上，加入少量额外的卷积层实现生成regions proposals，更快，效果更好

- 将一张H*W图片输入VGG，得到256维H/16*W/16的特征图。每个点生成k(=9)个锚框，将特征图输入RPN，经过3*3卷积层（输出仍为256维），分别经过两个1*1卷积层，输出为2k维H/16*W/16（softmax）和4k维H/16*W/16，分别储存了该锚框包含/不包含对象的概率、锚框位置的修正量，经过一个整合层（还会丢弃许多锚框）即可输出region propsals，这即为RPN。RPN输出的proposals经过Roi池化输入分类器
- 对特征图的每个点选择生成k个锚框（3种大小和比例），且RPN相当于k个分类器和回归器。通过这种方式，网络考虑不同尺寸的bbox
- RPN和Fast R-CNN共享特征，在训练时采用了4-Step Alternating Training方式，损失函数和Fast R-CNN类似

5.OHEM:Training region-based object detectors with online hard example mining(2016)

论文链接

在基于region的目标检测器中，前景和背景样例比例不平衡是一个挑战，但这不是一个新的挑战，在过去20年bootstrapping方法（hard negative mining）在传统目标检测任务中（如SVM）被使用。本文提出Online Hard Example Mining，提出boottrapping方法在线学习的形式，从而应用到CNN网络中，取得了很好的效果

- bootstrapping的思想大致为：训练时有两个交替进行的阶段，使用fixed model寻找hard example（假正例，或违反当前分类边界）并加入到active训练集中（删去分类正确的example）；使用fixed训练集训练model。
- 这样的bootstrapping不适用于SGD更新（在线学习）的深度卷积网络，会大大减慢速度，OHEM便是bootstrapping的在线学习形式
- OHEM的思想为：将一张图片输入CNN提取特征，计算所有RoI的损失，按照损失NMS后，选择损失最高的样例作为mini_batch进行反向传播（其他样例的损失置0）
- 当年的深度学习tool有限制，即使损失为0依然会占用内存反向传播，故本文提出一种较复杂的实现方式，加入了一个只读网络

6.Yolo v1:You only look once: Unified, real-time object detection(2016)

论文链接

与R-CNN系列不同，Yolo是one-stage目标检测，不会专门预测region proposals。本文提出Yolo v1，可以实现实时检测，但精度尚达不到sota，且对小目标误差较大

- 将image分为S*S个grid，每个grid生成B个bbox，每个bbox有五个参数（中心x，中心y，长，宽，置信度）。其中中心x和y坐标除以grid的长宽来归一化，长和宽通过除以image的长和宽来归一化。
- 置信度的计算：ground truth的置信度为P(该bbox所属的grid里有对象)*IoU。在训练时，将目标中心落在的grid看作有对象，P(该bbox所属的grid里有对象)=1，否则为0
- 训练时：包括四部分损失，中心点坐标损失（对于负责预测的bbox）+长宽损失（对于负责预测的bbox，且取平方根）+置信度误差（分别对于负责预测和无对象的grid，权重不同）+对象分类误差（对于有对象的grid）。对于每个grid，在训练时只选与ground truth IoU最大的bbox来responsible for预测，其余都算无对象
- 网络主体与GoogleNet类似，每张图片经过卷积层和全连接层后的的输出为S*S*(B*5+Class)形状的张量，直接预测每个bbox的五个参数，并预测每个grid含每个类别的概率，预测时对每个类通过NMS处理
- 计算长和宽的损失时取平方根，使小尺寸的bbox对长和宽的变化更敏感
- 预训练CNN时用224*224的图片作为输入，检测时用448*448更高分辨率的图片

7.SSD: Single Shot MultiBox Detector(2016)

论文链接

SSD也是一个one-stage目标检测方法，在不同尺度的特征图上生成先验框进行回归和分类

- 主要思想：将主干网络VGG的FC换为多个卷积层，选取多个（6个）卷积特征图，对每个cell生成k个（4-6）尺度和比例不同的默认框（即锚框），使用相互独立的卷积核（3*3）分别对不同特征图（设为m*n维）上的bbox进行分类（N+1类的置信度）和回归（offset），输出维度为(m*n*(k*(classes+4)))，将得到的所有输出（8k+预测框）进行NMS
- SSD采用多尺度特征图进行检测，可以识别不同尺寸的目标（有些方法是把图片处理成不同的大小，然后结合不同大小图片的结果），因为随卷积网络加深，特征图的维度下降，感受野增大，更low的特征图感受野小，有利于识别小尺寸目标。
- 和Yolo的主要区别：1.利用了不同尺度的特征图；2.使用了先验框 3.利用卷积核进行预测而非FC
- 训练时：利用文中提到的匹配策略确定正例，再对负样本进行hard negative mining采样，损失函数与Faster R-CNN类似
- 随特征图所属层数加深，默认框的尺寸增大，默认框可按比例对应到原图
- SSD还做了数据增强加强鲁棒性

8.R-FCN: Object Detection via Region-based Fully Convolutional Networks(2016)

论文链接

ResNet在图片分类任务大放异彩，本文想将FCN应用到目标检测任务，主要提出了一个既打破FCN平移不变性，又减少Rio-wise layer从而提速的方法，另外，R-FCN是应用了RPN的two-stage方法

- 背景：图像分类最新的sota网络如Resnet等使用了全卷积，因此自然地想将其应用到目标检测。但直接应用效果不好，因为卷积**translation invariance**，对平移不敏感，这有利于分类任务，而检测任务对对象的translation是敏感的。开始，作者试着将位置敏感的Roi pooling加入到卷积层之间，打破**translation invariance**，但这样会引入unshared Rio-wise layers，使计算效率变低，因此，提出了R-FCN

- R-FCN引入position-sensitive score maps使FCN对translation敏感，并且除了最后的position-sensitive Roi pooling（没有参数，很快），所有层都是shared，一张图只需前向传递一次，大大加速
- 在主干CNN提取出特征图后，一方面输入RPN生成Roi，另一方面利用3*3卷积，生成与特征图大小一致的$k^2(C+1)$个score map，即每个类有k^2张，每一张分别对应于一个grid（相当于将Roi分为k*k个grid）的得分分布。对于一个Roi，在score map中找到对应位置，对于每个类的k^2张特征图，分别其对应grid位置的score grid，拼成一个Rio score(与Roi形状相同)，共C+1个(**原论文图示中的颜色很重要！**)。最后，通过score投票，在文中简单的将Rio score加起来得到Rio的score，共得到C+1个score，用softmax可得分类概率。
- bbox回归与分类类似，区别在于score map有$4k^2$个，与之前的类似
- 训练时使用了OHEM，预测时使用了NMS

9.YOLO9000:Better, Faster, Stronger(2017)

论文链接

yolo v2在yolo的基础上加入了一些其他工作的先进idea和新idea来提高性能和速度，yolo9000在v2的基础上利用分类数据集联合训练，使模型可以识别对象的类别（即使不在检测数据集中）

- batch normalization，省去其他正则化和dropout
- 在224*224的分类集上预训练后，再用448*448的图像对分类任务微调，最后用448*448的图片训练检测任务，从而使预训练的模型更能适应高分辨率的图像
- 使用锚框并预测offset，使用k-means聚类来寻找锚框的最佳先验尺寸&比例
- 锚框预测offset的式子有所变化，将锚框中心限制在所属grid内
- 使用细粒度特征（多个特征图，如SSD）。top特征图为13*13，加入了一个paththrough层，让26*26的特征图拆成13*13并和top的特征图叠起来，共同作为提取的特征
- 因为Yolo v2只使用了卷积和池化层，所以训练时使用了不同尺寸的图片以提高鲁棒性
- 目标检测的数据集远小于分类，因此本文提出了一个使用两种数据集联合训练的方法。对于带检测标签的数据正常求损失，对分类标签的数据只求分类损失，两种数据按一定比例采样。
- 通过建立WordTree统一两种数据集的标签（根据WordNet确定不同标签的从属关系），对于树中某个节点（标签）的绝对概率，通过将其自身和到root的所有父类的条件概率相乘得到。每个结点的条件概率即给定父类的条件下的概率，通过对父节点所有子节点做softmax得到。因此，在训练时，对所有同义词（属于同一父类）做multiple softmax来获得每个结点的条件概率，再对计算得到的绝对概率求loss，因此每张图片会更新GT label和其所有父类对应的梯度；在测试时，从root开始，每次选择条件概率最大的子节点，相乘直到绝对概率**小于**某个阈值。

10.FPN:Feature pyramid networks for object detection(2017)

论文链接

本文提出了FPN(Feature Pyramid Network)，利用在利用多尺度特征图的同时为高分辨率的特征图加入语义信息，获得了更快更好的结果

- 背景：为了识别不同尺寸的目标，传统方法是通过输入不同尺寸的图片，但时间和内存消耗太大。近来，如SSD使用不同尺度的特征图进行预测，实现不同分辨率的预测，但一方面，SSD为避免使用low的特征图（可能是因为包含的语义特征太少了，或者说太局部了），从第四个卷积层开始用于预测，这使得其对小目标检测不理想，因为lower层的分辨率更高，对小目标检测很重要；另一方面即使从第四层开始，相比top层，分辨率高的层的语义信息更少，不利于检测小目标，且层与层之间有语义gap。为了建立在多个尺度都具有丰富语义特征的特征图金字塔，且快速，本文提出了FPN

- Bottom-up pathway:自底向上的通路即在CNN前向传递期间生成的，尺度不断减小的特征图（选取每次下采样前的特征图），除了第一层（因为太大了占用内存）。这些特征图，越top分辨率越低，语义信息越丰富
- Top-down pathway and lateral connections：在Bottom-up pathway的top卷积层经过1*1卷积生成特征金字塔的最top层，设置为d维，再自顶向下逐渐生成每个特征图（尺寸越来越大），生成的特征金字塔每层的形状均与Bottom-up pathway的相同（维度可能不同）。生成方式为：对toper层进行上采样（本文简单的使用最近邻），生成与lower层尺寸相同的d维特征图，再对Bottom-up pathway中相应尺寸的特征图进行1*1卷积，生成d维特征图再与上采样后的d维特征图直接相加，最后加一个3*3卷积生成最终的特征图
- 应用于RPN：对CNN主干应用RPN，生成特征金字塔，对每个尺度的特征图分别生成锚框（每种特征图对应一个尺度和多个比例），再接入RPN网络预测。**对于不同尺度的特征图，应用共享参数的卷积层进行预测和分类即可**，原因大抵为不同特征图共享语义
- 应用于Fast F-CNN：按照Roi的尺寸将其分给不同尺度的特征图负责预测，从而使更小的Roi分给更高分辨率；预测时，对每个特征图直接应用Roi pooling，后接两个非线性层进行预测和分类，且不同特征图预测头的参数共享

11.RetinaNet:Focal loss for dense object detection(2017)

论文链接

提出了一种新的分类损失函数Focal Loss，使hard example的影响增大，解决one-stage方法中正负例比例严重不均的问题，提出的 RetinaNet在保持one stage模型精度的同时超过了two stage模型的精度

- 背景：one stage模型具有更快和更简单的潜力，但精度不如two stage，作者认为主要原因是one stage的前景与背景样本比例严重失衡（有许多简单的背景样本，而two stage已通过RPN选出Roi），而简单的背景样本（预测分类概率很接近1）在标准交叉熵损失下仍有不可忽略的损失，因one stage中简单的负例很多，会导致模型学习效率差，且使模型退化。本文通过修改标准交叉熵损失，解决了正负例不均衡问题，实现了比过去启发式采样、OHEM等方法更好的结果
- Focal loss:简单来说，从标准交叉熵-lnx，转为$-\alpha(1-p)^{\gamma}log(p)$，使p趋向于0时（hard example），损失更大，而p趋向1时，损失更小，从而降低简单负例的影响
- RetinaNet：one stage，使用了FPN作为主干网络，使用锚框回归（训练时match正负例的阈值都宽松了）
- 值得注意的一点是，在RetinaNet初始化时，通过对最后一个全连接层的偏置b设置，使训练开始时，每个锚框被模型标为前景的概率为π(0.01)，从而使背景为简单样本，避免大量背景样本的巨大不稳定影响。（默认初始化的话，前景和背景的概率差不多都是0.5）

12.Mask r-cnn(2017)

论文链接

mask R-CNN在实现目标检测的同时加入实例语义分割任务，并且提出Roi Asign方法，实现了更好的结果

- 网络结构的基础为以Resnet+FPN的Faster R-CNN。在提取特征图后，一方面输入RPN获得Roi，另一方面使用Roi Asign对Roi对齐到固定尺寸的m*m特征图。在原有bbox回归和分类的基础上，加入了语义分割分支，使用FCN对m*m特征图计算mask（对每个像素计算C个类的sigmod，未使用softmax以避免类间的竞争），后者损失为平均二元交叉熵
- 因Roi pooling对Roi的pixel-pixel对齐不好，过程中进行了两次取整操作，而Roi Asign可保留浮点数，利用双线性插值避免了取整引入的误差，大大提高了精度
- 在训练时，对于语义分割分支，只对每个mask的GT类mask值计算损失

- 预测时，先通过检测分支预测出最高分的k个Roi，再对它们进行mask分支，将得到的m*m*C mask取检测分支中预测的类别的得分（维度为m*m），再resize成Roi size，以0.5阈值二值化为语义分割输出

> 对如何从m*m维度mask resize成Roi原尺度存疑，似乎是语义分割中的dense predict方法

13.Yolov3: An incremental improvement(2018)

论文链接

这不是一篇正式的论文，尽管挂在了arxiv上，而且引用量接近两万，似乎是一个技术报告。本文讲述了Yolo v3做出的更新，和一些实验结果。

- 在预测bbox，多预测了一项该bbox有目标的概率（得分）
- 在分类任务中，不再使用yolo v2中的multi softmax，而是用了独立的逻辑分类器，训练时使用了二分类交叉损失熵
- 借鉴了FPN和残差连接
- Focal loss无用，猜测是因为预测bbox的有目标的概率，起到了相同的效果，使许多简单背景样本不产生损失

14.DERT:End-to-end object detection with transformers(2020)

论文链接

本文通过二元匹配和transformer编码解码器结构实现DERT结构，大大简化了检测任务的pipeline，简化了许多人工设计（NMS,锚框），实现了集合预测（一次预测出所有对象的类别和位置）

- 结构（附录里的细节图片非常非常清晰！）：先用CNN提取特征，与固定的位置编码相加后输入transformer encoder;向decoder输入一组可训练的object query(应该是N个)，在cross-attention部分应用了encoder的输出，得到N个预测（N是一个固定值，明显大于一个图中可能的目标数量）；将N个预测输入FFN获得bbox预测和分类概率，包括"no object"类
- loss:
    - 训练时先将N个预测与GT做二元匹配（将GT用no object填充成N元组），即寻找一个排列，使预测和GT pair-wise的匹配成本加起来最低。y_i与y_{pre-i}匹配成本为：若y_i的类c_i为无对象则0，其余为$-p_{pre-i}(c_i)$+二者bbox的损失（下述）
    - 选出排列后，按预测与GT的二元匹配，计算loss，其中分类loss用-log，bbox损失使用L1 loss和generalized IoU loss（附录中有详细说明）的加权（避免只用L1 loss对尺寸敏感）
- 辅助loss：每个decoder块后都会接上FFN进行预测并计算损失，在输入FFN前加了一个不同层之间shared层归一化
- 并行解码：没有使用原始transformer中的auto-regressive解码，而是训练decoder的输入——一组object query，只在cross-attention中使用encoder的结果，实现了并行解码

Semantic Segmentation

1.FCN:Fully Convolutional Networks for Semantic Segmentation(2015)

论文链接

本文使用全卷积网络实现了pixel-pixel,端到端的语义分割模型，还提出了一种利用多尺度特征的方法

- 背景：分类网络CNN取得了很好的效果，想迁移到语义分割任务——去掉分类层、将FC换为conv、加入上采样实现dense predict。非线性层使CNN只能接受固定尺寸的输入，而每个FC可以等效为一个卷积层，因此使用FCN可以接受任意尺寸的输入。
- 分类任务中，卷积网络在提取特征的过程中会不断地下采样，使特征图尺寸不断下降，这使top层的特征分辨率较低不适应于pixel-wise的语义分割任务，需要让分类网络适应dense predict。本文检验了overFeat中提出的shift-and-stitch方法（没使用），最终使用了上采样方法——反卷积/双线性插值（最后一次上采样将反卷积初始化为双线性插值，再学习），和pixel loss实现了dense predict
- 结合高分辨率浅层和低分辨率高层的语义特征，FPN应该是对此有所借鉴。在对top层（第五层）上采样32倍时，FCN-8s将第五层先2倍上采样再与经过1*卷积的第四层相加，将结果2倍上采样，再与经过1*卷积的第三层相加，将结果8倍上采样得到与原图尺寸一致的输出，从而结合了多个尺度的特征图。（如果融合更low的层收益递减）
- top特征图的通道数为C（类别数），因此相当于特征图的每个点为C维张量（每个类的得分），信息太少了！不利于后面大尺度的上采样，这在U-net中进行了改进，在上采样部分仍保留了丰富的特征通道

2.U-Net: Convolutional Networks for Biomedical Image Segmentation(2015)

论文链接

这是一篇用于医学图像的语义分割论文，但提出的U-net是一个广泛取得优秀结果的模型

- U-net也采用了全卷积网络，与FCN相似。先前向传递一个CNN获得下采样的一系列特征图，将top层特征图经过两个3*3卷积层后，进行一系列上采样(*2)，每次上采样后，将结果与**下采样过程中对应的特征图**裁剪后拼在一起(concatenation)，经过两个3*3卷积层后进行下一次上采样，最后一次上采样后shiyong1*1卷积层后的每个像素的分类。上采样和下采样过程比较对称，形成一个U型结构（论文中的图片很清晰）
- 一个比较重要的点。U-net中绝大部分使用的是3*3卷积层，没有pad！所以每经过一次卷积层，特征图尺寸都会-2，因为这个原因上采样和下采样对应的特征图尺寸有所区别，需要将下采样的特征图裁剪后concatenation。文中认为在边缘pad会使边缘像素的特征随深度增加而越来越模糊，特征图尺寸下降也与下述overlap-tile策略有关
- 也许是医学图像的问题（分辨率太大），也可能是当时的设备限制（内存小），也可能是因为数据量小（切片增加数据量），U-net使用了overlap-tile策略，将图片切片成m*m的patch，并对patch进行padding（即取patch周围的上下文像素），使padding后的patch经过U-net后（尺寸会降低）尺寸恰为m*m。对于图片边缘的patch，可能有些方向没有上下文来padding，这时使用镜像padding，用patch作镜面对称。通过这种方式，可以实现对任意大图像进行无缝切割后进行预测，每个patch也获得了上下文信息。
- 与FCN在上采样有一个不同，FCN上采样时直接对分类分数上采样，显然很不准；U-net在上采样时保留丰富的特征，在最后才用1*1卷积层分类

> FCN在结合下采样特征图时将其1*1卷积后直接相加，U-net先concatenation再经过3*3卷积融合，FPN将其经过1*1卷积后相加再经过3*3卷积融合

- 为了提高对"接触的目标"的区分，本文使用了加权交叉熵损失，使用了一个公式（见论文），在训练前对每个GT图计算权重图，这种方法会使目标间的小背景具有较高的权重
- 医学图像分割任务的一个挑战为有标注数据很少，本文使用了数据增强，其中随即弹性形变的效果最好

3.Segnet: A deep convolutional encoder-decoder architecture for image segmentation(2016)

论文链接

网络结构与U-net类似，先下采样再上采样最后分类，提出了一种新的上采样方法，减小内存。虽然文章很长，但创新点有限

- Segnet的动机是实现高效的场景理解结构，更注重于优化时间和内存消耗，同时在各项指标上具有竞争力。
- SegNet应用了encoder-decoder结构（下采样和上采样阶段），encoder为FCN，卷积+BN+ReLU+最大池化得到该尺寸的特征图，decoder先上采样再接卷积层再BN再ReLU，最终实现像素级分类
- 关键点：在encoder中，只记录特征图max pooling时最大值的索引，从而使需要记录的特征信息大大降维。上采样时用了max pooling indices，根据encoder中对应特征图池化时的最大索引，实现上采样（对应索引取值，其余置零）。上采样后的特征图是稀疏的，后接三个（卷积层再BN再ReLU）得到稠密的特征图用于下一阶段的上采样。上采样不需要学习也提高了效率。
- 实验表明，使用全部encoder时的特征图可以得到最好的效果，但在内存受限时SegNet可以提高表现

4.PSPNet:Pyramid scene parsing network(2017)

论文链接

本文提出了应用了Pyramid pooling module的PSPNet，可以聚合不同区域的上下文特征，并加入了一个辅助loss来训练深度ResNet

- 背景：全局信息和上下文关系对场景分析（语义分割）是重要的，简单的使用全局池化会损失空间关系而导致歧义，因此提供了一种金字塔池化，从而建立全局场景的先验。
- 将图片输入主干网络得到top特征图，将其按照不同尺寸池化，池化后有N*N个bin(N=1,2,3,6)，N=1时便为最一般的全局池化，这样可以得到不同尺度子区域的representation，不同水平的上下文信息。对每个池化后的context representation，用一个1*1卷积层将N*N尺寸的维度降为1\N，从而保持个水平全局特征之间的权重。之后分别进行上采样（双线性插值），使尺寸恢复为原特征图大小，再将这四个与原特征图concatenation，进行卷积以得到最后预测
- 对于主干网络，使用了ResNet和扩张卷积，在训练时，除了对最后一层的特征图进行预测，还加入了一个辅助损失，在res4b22残差块进行预测，共同反向传播更新网络，帮助优化学习过程。（前者的权重更大）

> 感觉PSP和目标检测中的SPP思想基本一样

5.Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs(2017)

论文链接

本文提出了Deeplab v2，在v1的基础上改进，因为v1的论文没看，所以读的有些粗糙，一些细节没弄清楚，之后若用到再细研究

- Deeplab的主要特点为：应用了空洞卷积(atrous concolution)；使用ASPP模块(atrous spatial pyramid pooling)；使用了CRF(Conditional Random Field)，这个方法在后续版本被抛弃
- 背景：应用于分类任务的CNN建构对空间变换具有一定的鲁棒性，这对分割问题不利——降低了分辨率、处理不同尺度物体、定位精度下降，第一条的三个特点分别解决这三个挑战
- 空洞卷积：空洞卷积可以在保持特征图视野大小的同时矿大感受野。为了扩大感受野，过去会增加步长或池化，会降低特征图视野大小，本Deeplab应用了空洞卷积，将Resnet第五个池化层及之后的池化换为

步长为2的空洞卷积，从而由原来的下采样32倍变为下采样8倍。之后再用双线性插值上采样8倍，恢复原图像尺寸进行预测

> 空洞卷积可能导致grid problem，即感受野扩大，但某些最邻近的像素被忽略，可以通过连续使用不同尺寸的空洞卷积来时感受野铺满

- ASPP：在预测时，为了获得多尺度特征，对特征图进行了4个尺度下的空洞卷积，后分别又接了卷积层，将得到的4个输出和一个全局池化值（先全局池化再插值，细节不清楚）五部分concatenation起来，进行最后的预测

> 因为没时间读v1的论文了（大概也不太重要吧，而且现在是transformer时代了），可能一些细节没搞懂，以后再说

6.RefineNet: Multi-Path Refinement Networks for High-Resolution Semantic Segmentation(2017)

论文链接

本文也是为了解决下采样过程中导致的分辨率下降问题，提出RefineNet利用下采样过程中的所有信息来细化富含语义信息的top层特征图，且帮助其上采样，还提出一种链式残差池化

- 背景：分辨率下降是语义分割任务常见的挑战，一种方法为下采样后通过反卷积等方式上采样，但其实没有利用细粒度特征；另一种方法为Deeplab提出的空洞卷积，在保持视野大小的同时扩大感受野，但一方面会增加许多高维卷积占用很大内存，另一方面空洞卷积也是一种下采样，潜在的丢失一些信息。本文提出了RefineNet，使用下采样过程中的多尺度的、高分辨率特征图，细化帮助上采样时语义信息丰富、分辨率低的特征图，思想和FPN比较类似
- RefineNet下采样时使用的ResNet主体结构，利用了第二个池化层开始的特征图(1/4--1/32)。将1/32的特征图输入RefineNet4(这是一个block)，输出1/32的新特征图，再和1/16特征图一起输入RefineNet3，输出1/16的新特征图，依次下去，直到得到融合了细粒度特征的1/4特征图，做softmax再双线性插值
- RefineNet块里做了什么：先将1/2个特征图（对应Resnet块的特征图和上一个RefineNet块的输出）分别输入两个级联的RCU(残差卷积单元),每个RCU包括两个3*3卷积和ReLU和残差链接，其中除了RefineNet4的输出维度为512其余为256（RCU的目的是将预训练的适用于分类的特征图适应于分割任务，一种解释罢了）；将输出进行multi-resolution fusion，分别输入3*3卷积（将维度统一为最低的）和上采样（将尺寸统一为最大的），再相加；将输出进行Chained Residual Pooling，将输入进行级联的带残差链接的池化+卷积块，也就是每进行一次池化+卷积，都与这次的输入相加再输入到下一个池化+卷积（这样可以得到丰富的不同尺度的池化特征，并通过卷积权重加起来，认为这样可以有效捕捉背景上下文特征）；将输出通过一个RCU得到最终输出。
- 在整个网络中，应用了丰富的残差思想，既有短程（块内）的残差连接，又在上采样时与下采样时的特征图连接，是梯度更容易的传到靠前的参数中，有利于端对端训练

7.SERT:Rethinking Semantic Segmentation from a Sequence-to-Sequence Perspective with Transformers(2021)

论文链接

将纯transformer结构应用到语义分割任务，使用encoder-decoder架构，提出SETR，创新点不大

- 预处理：和ViT一样，先分成patch，再映射到patch embedding，加上position embedding作为输入
- encoder:24个tranformer encoder块，相应的有24张特征图
- decoder:语义分割的难点在于，将特征图的尺寸恢复到原图分辨率，本文提出了三种decoder方式

- Naive:将encoder最后一层特征图reshape成3D后，先用卷积层将维度转为类别数，在双线性插值到原尺寸
- PUP:将encoder最后一层特征图reshape成3D后，交替上采样*2和卷积层
- MLA(multi-Level feature Aggregation):和FPN类似，取M个encoder的特征图，先reshape成3D，再分别经过卷积层和4倍上采样，再加入一个横向连接，分别经过卷积层，再按维度concatenation,最后经过卷积层和4倍上采样得到原尺寸

小记

论文阅读效率：4 5 3 5 4 2

3.12周日开始读的，周二因为看《鹿鼎记》，效率偏低（），只读了两篇半（Yolo v3太短了），其他还是蛮可以的。

周一和周三都读了五篇，神奇的地方在于，这两天上午都是满课但是能读两篇，下午都是空课但都只读了一篇，晚上各读了两篇，从效率的角度来说也许是因为下午比较摸，所以晚上效率高？（笑

周四还是比较肝的，下午只读了一篇deeplab，因为直接读的v2读得不太顺畅；晚上花了三个多小时读了RefineNet，这文章写得重复和故作高深的地方不少，看得我昏昏欲睡，最后读懂了才发现创新点不大；之后读了ViT，图书馆闭馆后又去教学楼继续写完总结才回宿舍。回到宿舍趁热打铁继续把DERT读了，写到晚上一点半。

周四周五开始看另一部小说，虽然周四写到凌晨一点半，但看小说到三点多点才睡，导致周五上午虽然没课胆专注度有限，只读了一篇Swin，下午紧赶慢赶在体育课前读完SETR，到操场的时候已经快点完名了。

总的来说，遗憾在于原本计划最后读的一篇TransUNet来不及读了，周五实在太摸了，沉迷于看txgs，原本上午能读完两篇的；原本目标检测打算读RefineDet，这是自动化所雷震老师参与的一篇经典paper，但因为时间紧没读，其他感觉完成得还不错