PN++难变成各向异性 anisotropic: ball query&maxpooling 天然各向同性，需要 MLP field (3D2DCNN 天然各向异性) 3dCNN complexity issue 且 voxelization 会产生信息丢失—sparse voxel: 只储存表面的信息，把计算限制在表面附近 Sparse conv: 有一定程度的信息丢失，但大尺度上无影响，PN++在大尺度上开销太大。和 Conv 有类似的表达力和平移等变性，网格支持索引(邻居索引哈希速度快)。会有离散化误差 discretization error，因为转换成了 voxel。

Sparse Conv 分辨率受限，常用大尺度如激光雷达，小尺度如机器人用 Point cloud networks 精度更高，但是 FPS 和 Ball query 很慢

RNN: Image/Video captioning etc. ht=fw(ht-1, xt), yt=fw1(ht) 也可以选择不输出。Recurrence: 每次都是同一个更新公式和 weight(可处理任意长), vanilla rnn 用 tanh(Whh ht-1+Wxh xt). h0 常被初始为 0. 假设输出有 T 步，至少 1+⋯+T 个 grad 回流进 W，BP 计算极复杂。Truncated BP: 限制 sequence 长度，正反向均 T 步，看成 T 层的 NN。Hidden state 在前向时一直带着(感受野是之前所有的 x)，但是只 bp 一个 trunk（也可以双向 RNN，但是舍弃因果，无法由上文生成下文）。代价是所有训练只见过时间序列 T 的数据，前后因果关联短程。从 one-hot 到 embedded：前者对词太大太稀疏，后者一般是 pretrained。

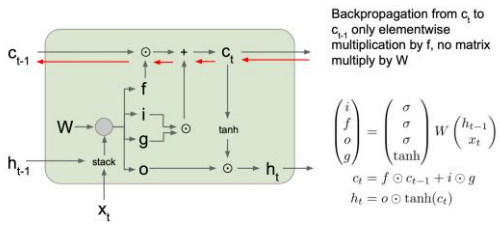
Exhaustive search: 穷举找生成概率最高的一句话 O(V^T), V 是字典大小。Beam search: 在每个 timestep 从 k^2 中选 k 个最可能的继续，k 是 beam size。Beam search 不保证最优，但是效率更高。Greedy sampling: 只选 prob 最高，问题是 deterministic，而 weighted sampling 可以生成更多样的序列，但是可能会 take wrong token. 最后一词感受野是前面所有词，但不代表都记下来了：

$\frac{\partial L_T}{\partial W} = \frac{\partial L_T}{\partial h_T}(\prod_{t=2}^{T} tanh'(W_{hh}h_{t-1} + W_{xh}x_t))W_{hh}^{T-1}\frac{\partial h_1}{\partial W}$ 几乎一定<1，梯度消失，较远的信息消失，缺少 long-term effects。不用非线性部分不解决问题，多个矩阵连乘奇异值>1 则会梯度爆炸，用 gradient clipping，<1 的梯度消失只能通过改变 RNN 的架构解决，比如用 skip link。

LSTM：每个 timestep 有一个 hidden state 和一个 cell state 储存长期信息，四个动态的 gate 在（以上均 n 维），gate 每一维都 0~1. Ct= f·c_{t-1}+i·g. g: how much (what) to write, tanh(其余 sigmoid)。I: whether to write, f: whether to erase, o: how much to reveal cell: ht=o·tanh(ct). 若 f 全 1，那么上面就是一个 skip link. 长程记忆有 uninterrupted gradient flow，但是短程仍经过 W。

LSTM 不保证一定没有梯度爆炸和消失，但是给模型一个更简单的学习长距离关系的方法。

GRU 和 LSTM 一样，他们的加法运算提升了 grad flow



Seq2Seq 是一种 many2many 但是先看完再生成，采用 encoder-decoder 架构，信息存储在 context vector 里，但是 c 大小固定，信息瓶颈。从最后 ht 预测 c 和 s0（decoder ini state），通常 c 即 ht。想在 decoder 每一步都用个新的 c 即 attention。Decoder $s_t = g_u(y_{t-1}, s_{t-1}, c)$ y0 是[START]，输出以[STOP]结束

多模态 multi-modal: image captioning CNN+RNN，可以 pretrain 一个 freeze 的 CNN 只训 RNN，也可以打开一部分 CNN; VQA: visual question answering: CNN+双向 LSTM，然后把各自 FC 的输出聚合，可用+，concat，element-wise. 用 ablation study 消融实验来证明你的 contribution。可以只 BP 聚合后的 FC，freeze 前面，让 bp 轻量化。多模态一般更加 expensive。

VQA 是一个合理但无价值的架构

每个 s_{t-1} 都回去和 hi 过一个 MLP 算 alignment score，再统一 softmax 成 attention weights。再用 **attention 为权对所有 hi 加权平均得到 Ct**，即 St 的 context vector。整个链路是**端到端可导的**，无需监督，bp 即可自动学习。

好处：输入序列不再受单一 vector 瓶颈；context 可以在每一步去看输入序列的不同的相关部分。Decoder 在回传过 MLP 的时候不把 hi 视为有序序列。

Attention 的核心愿望和 U-Net 一脉相承，但是不能是一一对应，而是加权。

在当前流程中，decoder 的 states 是 query，encoder 的 states 是 key，context states 是 output。过程抽象为：输入 query vector[D_Q]和 data vectors[N_x x D_x]算相似度[Nx]再 softmax，最后加权输出[Dx]。相似度可以用点积算，但是高维向量 dot 可能很大，softmax 可能饱和导致梯度爆炸，所以要除以根号 Dx.

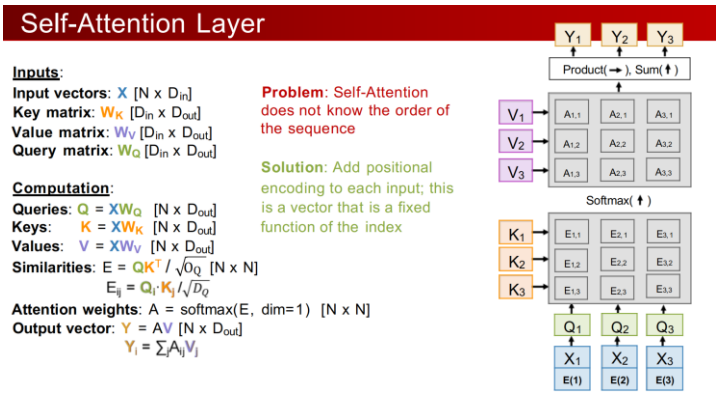
进一步：用多个 query vectors（矩阵），把 data 的 X 解耦成 Key 和 Value

这是 Cross-attention，KV 来自 X，Q 是输入。$E=QK^T/\sqrt{Dk}$, A=softmax(E,dim=1) 竖着，Y=AV 横着乘，竖着加。每个 Y 都是 V 的加权和。

排列 X，输出不变（应该等变才对），排列 Q 输出等变。

Self-attention 增加了 W_Q，Q 也来自 X，输出**对于 X 等变**。

VS 一维卷积：1Dconv 的感受野受限于卷积核长度，而 self-attetion 全局视野



可用 masked self-attention 保证因果性 causal（变成-inf）但仍是平行运算

多头：先投影再拼接。实践中用 batched matrix multiply 来平行运算所有头。多头的表达力更强，因为每个头有自己的注意力，关注的也是不同的 V

Self-attention 是四个矩阵乘(highly Parallel，而且全局视野)：**(QKV Proj)** [NxD][Dx3HD_H]=>[Nx3HD_H], 3 是因为有 QKV, 分开得 QKV 大小都是[HxND_H]。**(QK Similarity)**E 是对应头的 QK 点积。**(V-Weighting)**得到[HxND_H] reshape [N x HD_H]. **(Output Proj)** 用一个 Output matrix[HD_HxD]把 Y 变到 O[NxD]，H 是头数. 中间两步涉及 NxN，如果输入长度极大那么 attention weights 就极大。

解决：Flash attention 合并计算二三步，不储存整个 attention 矩阵，可从 $O(N^2)$ 降到 O(N).

Transformer block：input->self-attention->reslink->layer norm->每个 vector 独立过 MLP(FFN)->reslink->Layer norm->output. LN 和 MLP 都是在每个向量维度上的，LN 算一个 token 在 D 个 channel 上的平均值。Vector 之间的交互只有 self-attention。整个过程只有 self-attention 的四个加 MLP 的两个共 6 个矩阵乘，highly scalable & parallelizable。垒叠 transformer block 即可。

LLM 会在输入的一开始过一个 embedding matrix，把 vocab size 转成 D 维 input，最后学一个 projection matrix 再投影到 vocab 空间输出

VisionTransformers(ViT)：把图片分割成 patch，直接对每个 flatten+linear，可被视为 16x16conv stride=16 3 to D channels，加上位置信息再一起扔进 transformer 最后给出 global pooling 后的结果。Visual token 不需要 mask。

一些改进：Pre-Norm，把 LN 放到 self-attention 前，不然 LN 不在 res link 范围内，使训练更稳定；RMSNorm root mean square，一个可学习参数γ，经过 RMSNorm 后数据的 rms 就变成γ; SwiGLU MLP。Mixture of Experts (MoE)：学 E 个 MLP 称为 E 个 expert，有一个门控（不可导），对每个 token 会经过 A<E 个 expert，称为 active experts。A 控制前向计算量，但总参数量大大增加。

Semantic segmentation：不关心 instance，只关心类别；Object detection：区分每个 instance，但轮廓不精细；Instance segmentation：实例分割

Object detection: localization + classification. 输出为(axis aligned 的)bounding box（在 3D 里可能空隙太太）。Tight BBox：所有属于它的 pixel 均被包含且不能更小。2DBBox 有四个自由度，定义参数常用 x,y,h,w，左上角坐标加长宽。

对于 single object 的网络，可以设计 classification branch 和 localization branch，把后者视为一个回归问题用 L2loss。也可加上前者的 softmax 组合成 multitask loss。X, y, h, w 都有范围，加个 sigmoid 控制输出范畴。

Regression loss 看图像：L1（绝对值和）很鲁棒，不管 loss 多大 gradient 与其无关，但不能在 0 处很好停留；L2loss（平方和，和 L2 norm 不同）初始 loss 可能过大，对大 error 不鲁棒，但是 convergence 好，通常的简单选择。

此外还有 RMS(见上)，用得少，在 0 处梯度发散。**Smooth L1** 是一种改进。

Multi-object：用滑动窗口扫描太昂贵，要做无数个 CNN—>region proposal

R-CNN: 1. Propose Regions of Interest(~2k) 2. 全部 Warp(扭曲）成固定大小(224x224) 3. 输入在 ImageNet 上预训好的 CNN，移除分类层，把最后一个 FC 层的特征向量当输出 3. 用 SVM 分类 4. Bbox 回归

问题：1. 需要做~2k 次独立的 forward，非常慢 2. Cropped region 失去了其 visual context，不能知道要向外扩多少，不能为 Bbox reg 提供足够信息

改进：crop 前输入 CNN，改为 crop conv feature

Fast R-CNN：对全图进行一次 Conv，在 feature map 上 propose RoI，此时每个 pixel 已有较大的 receptive field。Propose 依然是基于原图，但是因为 feature map 的 resolution 变小了，要把 RoI 做相应变化。1. RoI 要'Snap' to grid cells 吸附到 feature map 新的 resolution 网格上(保证 axis align，同时要吸附到最接近 heuristic)。2. Resize（**RoI pooling**）：roughly divide 成 2x2（或其他），然后 max-pool

Fast R-CNN 训练速度提升了 10 倍。RCNN propose 2s，forward 47s，而 Fast 的 forward 只要 0.32s，主要瓶颈已经变成了 region proposals (希望能 1s 做几十次检测，就可以做到追踪)

Faster R-CNN: region proposal network (RPN) 作用于 feature map。用一个固定大小的 anchor box 做滑动窗口（feature map resolution 低，所以可以这么做，而且因为每个 pixel 都包含周遭信息，只需要 fixed size，再根据实际缩小即可）在每个特征图位置上会放 K 个锚点框(以当前 cell 为中心换不同的 H 和 W)，共 KxHxW 个，每个锚点框要进行尺寸和位置的调整。这 KxHxW 个称为 raw region proposal，对它们进行二分类算 objectness（是物体的概率），然后根据这个排序选 top300 作为 RoI 交给 RoIpooling （实际训练中会出现正负样本极不平衡的情况，要对 300k 个 raw proposal 做一定的筛选凑一个比例）

Faster R-CNN 要共同训练四个 loss：RPN 的 objectness（二分类），RPN 的 box reg 坐标（和 ground truth 算 IoU），最终的 n 分类，最终的 Bbox reg 坐标

Faster R-CNN 能做到 3~5 frames per sec。two stage 当中的过程（如选 top300）是不可导的，不能端到端训练，而且很冗余，希望能把两个 stage 合并成一个。

Single-stage detectors：**YOLO**/SSD/RetinaNet. 直接把原图分成 7x7grid，对每个 grid cell 选 B 个 base box 做 5 number reg（dx，dy，dh，dw，confidence），同时做 N+1 类分类（包括背景），output：7x7x(5*B+N+1)。Box 选的少导致不够精准，但特别快，能做到 100~200 fps。

NMS 采用 IoU 做 threshold（先找一个类别中最高 confidence 的 box，然后看其他 box，IoU 大说明和已有的重合高，被视为冗余 suppress），对每个物体留下一个最好的 box，无法端到端训

对整个模型最 naïve 的评价是用 classification%代替 Bbox%，若训练合理，更好的 bounding 的确能预测出更好的准确率；评估方法：**Average Precision**：Precision-Recall curve。Precision=TP/TP+FP, R=TP/TP+FN(漏检）背后变量是把所有 box 按 confidence 从高到低一个一个增加，看 PR 怎么变。11 点法算 AP：R 从 0~1.取 11 个点给 P 取平均。能关于 IoU Thre 做不同的 AP，不同类也可以

Instance segmentation：Top-down 先 object detection 然后再找一个 mask；Bottom-up：先生成 mask 再做分类。找 mask 称为 grouping，看 pixel 和周围是否属于同一个物体。

Mask R-CNN：top down，把 RoI pool 改成 RoI align，取消了吸附。吸附后在把 Bbox 还原到原图的时候会有一个不可知的偏差，导致 mask 和物体边缘对不齐。改成用双线性插值，保留了原始信息，可返回。对每个实例都生成 C 个 mask，形成了一个先验知识，最后用分类头选择用哪个 mask，结果质量更高。RoI align 在 AP75 上增幅显著，因为 thre 高就对精细度有高要求，bbox 的 refinement 也得到了提高，因为 ground truth 是根据 snap 前给的，snap 却引入了一个不可知的噪声。

生成式模型：学一个 $P_{model}(X)$近似 $P_{data}(X)$，并从前者 sample 新的 X. Formulate as density estimation problem，可以显示定义和解出 Pmodel 也可以隐式。

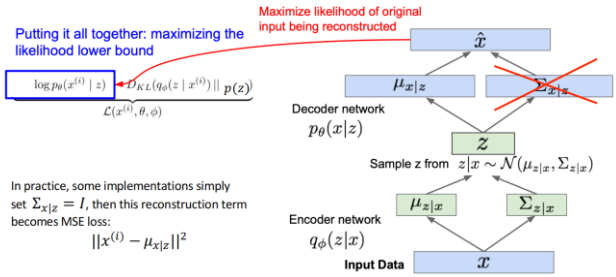
Discriminative model 学的是 X inputs 下的条件概率，而 Generative 学的是全空间里的概率分布，这个分布及其复杂。Explicit 又分为 Tractable 和 approx。

Explicit density model：Fully Visible Belief Network (FVBN)，利用概率的链式法则将高维数据 X 的联合分布分解为条件概率乘积 $p(x)=\Pi p(xi \mid x1,x2,...,xi-1)$，按 pixel 依次生成，后生成的以前序所有为条件，可以是 transformer/CNN. 训练目标直接使用 MLE。顺序生成速度慢，建模方式不自然。PixelRNN、PixelCNN

Variational Autoencoders（VAE）变分自编码器：approximate density mode

VAE 本质是一个概率的 Autoencoder，区别于 deterministic：输入对应唯一输出。Autoencoder 是自监督的，输入数据直接作为监督目标。

全空间大小是 224x224x3，数据在其中只是极薄的 manifold 流形，占据概率极低。Autoencoder 不是生成式本质是不知道 Zdata、没有 Zmodel，无法从中采样（如果强行从 Z 空间随机采样会解码出无意义的噪声）。于是我们必须假设 Z 满足一个可采样的分布，VAE 假设为标准正态高斯分布，符合一般分布直觉。（也有问题，不一定是单峰的）我们希望 q(Z)尽可能像 p(Z|X)要求的后验分布 Likelihood 是一个积分算不出，用蒙特卡洛估计期望？对生成式任务噪音太大。把 z|x 概率近似成 encoder，把 likelihood 展开，最后一项 KL 散度算不出但恒正，前面两项组成 ELBO，是 likelihood 的可优化的 lower bound。任务转变为 maximize ELBO。端到端可导。网络给出 Z 和 Xhat 的均值和方差来确定分布。可以简单理解成，在 Autoencoder 的 L2loss 上再加了一项 encode 完的分布与标准正态分布算 KL 散度，这一项要尽可能小，控制 Z 空间的分布相似度。



这两项自身就是矛盾的，第一项要求 X 与标准正态足够不一样，因此不可能训练到底，生成的图片一定是模糊的，带有高斯噪声。

GAN：生成对抗式网络。用一个神经网络 discriminator 来算打分 loss，对显示世界中不出现的 eg 高斯模糊有很好的分辨，L2 就无法做到。

Discriminator 二分类，所有真图 label 真，所有生成图 label 都是假。和 generator 是博弈互相增进的关系，但优化的是两个参数而非同一个参数，所以可行。最终形成一种纳什均衡。D 任务更简单，会很快变得很强，使 G 在一开始优化不动 grad 太小。在一开始把 G 的目标改成升自己是真图的概率（本来是降自己是假图的概率）。GAN 的隐空间里涌现出了线性结构。问题：**分布 cover 不好**。

评估 GAN：FID: 把一堆真实图片和一堆生成图片输入一个无关的神经网络，取中间某一层的输出作为特征向量。把所有两组图的特征向量分别合在一起作为两个个高斯分布算距离。FID 越小代表两个分布越相似。