今天和大家分享一下transformer以及它在CV领域的应用。主要从以下四个方面来介绍，首先是transformer提出时的研究背景，在这里也会简单介绍一下它从RNN的发展，接着是transformer的网络结构，然后是transformer在CV领域的应用，最后对transformer最近一两年在CV领域的发展做一个展望。

首先先来讲一下，transformer提出的背景，自然语言处理简称NLP，主要是通过一些算法对我们日常生活中使用的自然语言进行分析。这个自然语言包括说出来的和写出来的。比如现在有人说了一句英语，那我想知道他说了啥，这就是机器翻译；然后我想回应他说的话，这就是智能问答；之后我还想预测一下它接下来要说什么，那就是文本预测或者叫文本续写；最后我想分析一下它是在骂我还是夸我，那这就叫情感分析。自然语言处理的主要流程是这3个步骤，其中文本预处理相当于数据清洗，比如现在给一句话，不能直接对一句话进行处理，而是要先把这句话分为一个一个的单词，这个过程就是预处理。那分为单独的词语以后，还是不能直接用啊，因为网络在计算机上训练的时候，需要给他提供向量，提供矩阵，这就是文本表示的功能，把之前分割好的词语用向量表示出来，这也被称之为词嵌入（word embedding），将词语嵌入到低维的向量空间。

因为后面网络会用到这个概念，所以我们在这专门说一下词嵌入。词嵌入刚才已经提到了，就是把文本单词映射为一个几百维或者是几千维的向量，在映射的过程中会考虑它周围的单词和上下文的信息。目前词嵌入的常用算法有这几个。我拿其中的spaCy库演示一个简单的例子：巴拉巴拉介绍一下例子。可以再提一下独热编码。那得到词向量以后就可以进行真正的计算分析了。

接下来简略讲一下transformer的发展。首先是RNN，因为之前看到经过embedding以后，得到的是一个向量，也就是一维数组，RNN非常适合对这种序列数据进行建模，RNN的结构就不用多说了，这张图展示的比较清楚，当前的输出由当前的输入和上一个状态的输出决定。但是RNN的缺点也很明显，一个是当序列很长时，RNN会丧失对之前序列的学习能力，导致梯度消失或梯度爆炸的现象；另一个就是RNN输入对应的输出是定长的，这对于很多任务来说，局限性都很大。那为了解决RNN的长序列依赖问题，就有人提出了LSTM，通过加入一些“门”的网络结构设计，有选择的遗忘一些之前记忆的信息，实现了长序列数据的稳定学习。这个结构就不做详细解读，因为比较复杂，有很多论文也在研究改进这个，有兴趣可以看看。

然后是上面提到的RNN的第二个问题，有人提出了seq2seq网络。通过这种编码-解码器结构，实现了m到n的序列输出。这个编码器和解码器可以用rnn和lstm。这是他内部运行结构。但是这里seq2seq还有一个问题就是，这里的C是输入的语义编码，并通过C得到的输出，那这相当于对输入的单词一视同仁了，也就是对于输出来说，输入的贡献是相同的，这显然不太符合实际情况。注意力机制作为解决这一问题的方法被提出。这是引入注意力机制的seq2seq结构，它实际上就是通过对输入进行加权，得到不同的特征C，每个C对应一个输出。所以注意力机制本身是一种加权机制。

接下来就是transformer了，Transformer其实也可以用带有注意力的编码-解码器结构来描述。但是它的编码器和解码器抛弃了RNN或LSTM的结构，只剩下了自我注意力和前馈神经网络（也就是线性层），那这样设计的好处有什么呢，一个很明显的好处就是，相比之前使用的RNN等串行计算结构，transformer可以在编码器与解码器中实现并行计算。这个接下来咱们在讲它的网络结构时细说。

接下来讲一下transformer的网络结构。把编码器和解码器拆开来讲，因为解码器结构和编码器区别不是很大。所以先来看编码器。（把编码器流程描述一下）

首先是将输入的单词embedding，变为词向量，注意这里还对每个词在句子中的位置进行了位置编码（pos embedding），进行位置编码的原因是，之前用的RNN是串行运算的，天然的带有序列性，而现在改为并行计算以后，得不到位置信息了，所以需要对其进行编码。根据位置编码的这两个原则，原文中设计了这个sin、cos交替编码的方式，关于这个公式为什么满足了以上两点要求，解释起来比较复杂，感兴趣的话可以看这篇文章，有详细的公式推导。

得到embedding的向量以后，经过一个多头注意力层和Norm层，然后再经过一个前馈神经网络层和Norm层，前馈神经网络就是咱们经常说的线性层，注意这里还用了两次残差连接，这个就不多说了，还有就是它这里采用的Norm方法是LN，与咱们经常用的BN不一样，BN是对每个批次的样本进行归一化，LN是对每个单独样本进行归一化。

所以大家可以看一下，这个网络的组件中，很多都是比较熟悉的，最后只有一个多头自注意力层，是比较重要的。我们来看一下计算自注意力的过程。

首先，先初始化一组WK，WQ,WV的权重矩阵，这就相当于卷积网络中的卷积核，里面的权重可以通过反向传播来更新。然后（把这个矩阵相乘的过程描述一遍）。得到K，Q，V矩阵以后，我们再根据这个公式进行注意力计算，注意这里计算时，是每两个词（包括自己）之间进行一次计算，具体来说，就是（把图片描述一遍，讲一下每个单词的K，Q，V之间是怎么相乘的。）

捋一遍这个过程，就单词首先进行embedding变为向量，然后初始化权重WK，WQ，WV矩阵，向量与矩阵相乘，得到K，Q，V矩阵，然后单词两两之间运行求注意力，最终得到结果。

好的，自我注意力的运算流程算是讲完了，那么这个自我注意力实质上是在干嘛呢，这里简单描述一下。自注意力的核心就是从大量的源信息中，找到对当前任务有用的重要信息。像这幅图表示的这样，每个单词都会通过自注意力的计算，得到其他单词与自己的相关程度。（稍微把图描述一下）那这个运算结果是通过k，q，v的权值矩阵得到的，权值矩阵可以通过训练不断更新，所以说，自注意力层实质上，是在学习如何关注与任务结果最相关的特征，也就是说，我要输出与真实值最接近的结果，我需要更关注那些特征，那些特征对于得到这个结果是更重要的。

那这样一想，这个可以用在计算机视觉是没啥问题的。比如在图像分类里面，你把标签看成真实值，把图像的特征值看做一个个的词向量，那网络相当于在学习那些特征对于分类准确是重要的，需要重点关注哪些特征。这幅图是VIT网络学习到的注意力图，可以看到它比较准确的关注到了有用特征，而忽略了背景。

之后还有一个就是多头注意力。多头注意力的运算过程和前面讲的没有什么区别，那有多个头，就要初始化多个权重矩阵。然后为了减少计算量，作者先将k，q，v矩阵通过线性层减少维度，再分别计算注意力，最后把结果堆叠在一起。这个其实和resnet里面的瓶颈层很像。

使用多头注意力就是为了希望网络能关注到更多的信息，这两幅图分别是不同注意力头运算的结果，可以看到，左边的倾向于关注全局信息，而右边的更关注一些局部信息。

最后就是看下编码器，编码器和解码器很像，所以只需要注意一下这两个区别就可以了，一个就是Mask层，Mask部分照着ppt读就可以了。注意第二个多头注意力层的输入，是由编码器输出的K，V值和上一层输出的Q值。这里的本质就是寻找与当前输出最有关系的词向量，得到下一个输出。

这里用动画演示编码器和解码器的工作流程。

————————————————上篇结束——————————————

下篇主要是讲一下transformer在计算机视觉的应用与发展，在应用这里，首先说一下transformer和CNN的联系与区别，然后和大家分享一下transformer在图像分类、目标检测和姿态估计这3个领域的应用实例。

那先来看一下，为什么transformer结构能代替卷积用在计算机视觉当中：

先看一下理论依据。人工智能领域有一种流形算法的假设，这一假设表明，无论是视觉图像还是语言，其实都是由基本语素组合构造的高维空间的子流形，无论是视觉还是NLP， 都是要找到这个子空间并且完成度量学习, 从这一点看，二者没有本质的区别，transformer可以用于NLP，也当然可以用于vision.

接下来对CV领域一直在使用的CNN与transformer做一个比较，CNN是由局部到整体的一个过程，而transformer则是直接获取到全文信息，然后利用注意力机制找到需要的任务目标。从长远来看，我认为将这两个结合进行使用，应当是一个未来的发展方向。

这是目前transformer在一些CV领域的应用，可以看到，基本上能想到的应用方向已经都有人在尝试了（当然还有一些比较小的方向应该是还没人做，比如水下的图像处理，缺陷检测等）

接下来介绍一种基于transformer的图像分类网络-ViT。一般来说，当前在CV中使用transformer有两种办法，一种是将transformer与CNN结合，还有一种就是ViT这种，完全抛弃卷积，只用transformer构建网络。

可以看到，因为transformer需要输入的是序列，所以这里首先把图像拆分为小块，然后把每个小块展开成向量，就是把N,H,W,C变为N,H\*W\*C，然后经过线性层进行embedding，并加上位置编码。需要注意的是，这里人为设置了一个可学习的类别的编码向量。这也好理解，因为VIT只用了transformer的编码器，在编码器最后，每个图像块都会有一个输出，对于分类任务而言，我应该取哪个输出向量进行后续分类呢？感觉哪个都不太好，那就专门设置一个用来分类的类别向量，用他最后的输出来分类。这个类别向量有点类似于编码器中Q矩阵，也就是说，VIT网络实际上是在学习，要准确输出类别标签，网络需要更多的关注图像中的那些特征。

然后介绍一下最先把transformer用在目标检测的DETR。可以看到，DETR在前面用到了CNN主干提取特征，然后编码器-解码器都用到了。这里的transformer和之前的网络有3个不同之处，我就简略讲讲。首先就是位置编码，DETR考虑了x与y两个方向的位置编码，这也比较符合实际图像的情况。还有就是编码器的输入，可以看到这里写的是object queries，这个就有点类似于VIT里面带的class头，这里相当于设置一个可学习的对象查询向量。网络通过训练学习到对象的信息并保存起来，在测试时，使用Q与测试图像的K，V进行注意力运算，实质上就是在查询有无对应的对象信息。最后的不同之处是，编码器的输出是一次性全部输出的，不同于原本transformer一个一个的输出方式。（这里作者是将目标检测问题看做集合预测问题，最终是使用匈牙利算法进行集合元素匹配。）

最后说一下最先使用transformer做基于回归的姿态估计网络PRTR。这个网络是在上面DETR的基础上改进的。因为在姿态估计中，很多方法都是先将人检测并从原图中裁剪出来，然后在裁剪的图像中，检测关键点。所以PRTR在网络的第一级使用了DETR来检测人和其他类别。检测到人并裁剪以后，送入第二级transformer网络当中，这个也是在DETR上面修改的，区别就在输出那里，DETR输出的是对象的框的坐标集合，这个输出的是关键点坐标的集合，然后DETR会输出类别，这个也会对关键点进行分类输出。

在最后一部分，就简要介绍一下，我所了解的transformer的一些发展，事实上，目前针对transformer的改进在不断进行着，就可能隔个几天就会出现新的改进，所以这一部分要深入了解的话，还是需要去多追一些比较新的研究。

首先是关于transformer架构方面的改进。这个讲一下微软前段时间的工作：swin transformer。先看一下SWT和VIT的对比，SWT将图片分成了一个个的窗口，这里将每个窗口看做一张图片，在窗口内部计算自注意力，然后在多级级联网络当中，逐步合并各个窗口，可以看一下这是网络的整体架构图。这个思路就是相当于卷积网络当中的先对图像局部进行卷积运算，然后通过卷积不断的对特征图进行下采样，提升感受野。所以这个网络虽然网络组件还用的是transformer的组件，但是思想却是卷积网络架构的思想。那这样做的好处是什么呢，一方面只在窗口算局部自注意力可以大大减少计算量，另一方面这样逐步减少图像分辨率，相当于获得了丰富的多尺度信息。最后看一下transformer块的内部结构，可以看到它是由窗口注意力模块和滑动窗口注意力模块交替进行的，滑动窗口注意力模块的作用是进行各个窗口之间的信息交换。

另一个发展方向是对transformer所使用的注意力进行改进。实际上，自我注意力只是注意力机制的一种。在研究如何提高注意力效果上有很多人在努力，这里介绍最近的一篇文章：VOLO。主要说一下它提出的outlook注意力机制，其他的一些细节不多赘述。Outlook就是认为之前VIT里面将一幅图分为16个paths，还是分的太大了，这会导致很多比较精细的特征得不到很好的表达，所以它就将网络分成了更多的paths，然后以每个paths为中心，计算它周围k×k个path的注意力，这相当于每个path都聚集到了它周围k×k个path的信息，然后outlook再把这个信息分发给周围k×k个path，相当于实现了细粒度级别的信息交换。然后在送到正常的transformer网络中去计算全局注意力，但此时的path中已经融合了精细级别的特征和大量的上下文信息。所以会产生更好的结果。

其实发展这里可以加上轻量化网络结构的方向。

最后总结：这里只是简要介绍这两个发展方向，事实上，包括上面讲的应用领域、网络结构与发展，都只能算是蜻蜓点水，transformer在20年与21年这两年得到了迅猛的发展，许多新的工作如雨后春笋，不断冒出。本次汇报仅做抛砖引玉之用，故到此结束。感谢诸位倾听。