经过了上一周的讨论，我们对于上周提出的两个问题进行了解决，并且对于论文的“Ralated Work”部分进行了阅读与相关内容的学习。

首先是关于attetion机制的了解。这个名字很明显其借鉴了人类的注意力机制，视觉注意力机制是人类大脑的一种天生的能力。当我们看到一幅图片时，先是快速扫过图片，然后锁定需要重点关注的目标区域。比如当我们观察上述图片时，根据实验，发现人们很容易将注意力很容易就集中在了人脸、文章标题和文章首句等位置。那么当我们用CNN 模型识别图像时，一般是通过卷积核去提取图像的局部信息，然而，每个局部信息对图像能否被正确识别的影响力是不同的，为了让模型知道图像中不同局部信息的重要性引入了注意力机制。他的目的在于简化模型并加速计算。

我们可以这样来看待Attention机制，将Source中的构成元素想象成是由一系列的<Key Value>数据对构成，此时给定Target中的某个元素Query，通过计算Query和各个Key的相似性或者相关性，得到每个Key对应Value的权重系数，然后对Value进行加权求和，即得到了最终的Attention数值。所以本质上Attention机制是对Source中元素的Value值进行加权求和，而Query和Key用来计算对应Value的权重系数。因此可以将其本质思想改写为如下公式。其中，Lx是Source的长度。

可以用一个例子来帮助理解。假设有个图书馆里有很多藏书，图书馆便是source，其中的藏书就是value，为了方便查找，给每本书都做了编号，即key，这样相当于我们在source中就拥有了一系列的KV数据对。我们想要了解鲁迅，也就是query，我们可以看他的传记，他写的文章，甚至中国近代史。为了提高效率，不会每本书都仔细阅读，对于传记和他写的文章会重点阅读，也就是权重较高；而近代史部分则会粗读，权重较低。因此，Attention机制可以理解为从大量信息中有选择地筛选出少量重要信息并聚焦到这些重要信息上，忽略大多不重要的信息。聚焦的过程体现在权重系数的计算上，权重越大越聚焦于其对应的Value值上，即权重代表了信息的重要性，而Value是其对应的信息。

对于Attention机制的具体计算过程，可以将其归纳为3个过程，如下图所示。第一个阶段根据Query和Key计算两者的相似性，第二个阶段对第一阶段的原始分值进行归一化处理，第三个过程根据权重系数对Value进行加权求和。在第一个阶段中，可以通过引入不同计算机制求相似性，比如两者的向量点积或是引入新的神经网络来自动学习。在第一阶段中根据不同方法产生的分值取值范围也不同，因此在第二阶段引入类似softmax的计算方法进行数值转换，一方面是为了归一化，将原始计算分值整理成权重和为1的概率分布；另一方面，通过softmax可以更加突出重要元素的权重。第二阶段计算结果ai对应valuei的权重系数，加权求和后可以得到attention数值。

第二个问题是关于conditioning 是怎么加入到运算的。作者是根据拓展得到一个条件时序去噪自编码器来实现的。通过输入条件y，如文本、语义图或其他图像到图像的翻译任务来控制合成过程。为了能从不同模态预处理条件y，论文引用了一个领域专用编码器，能够将y处理成一个中间表示，来帮助我们方便的引入各种形式的条件。最终模型就可以通过一个cross-attention层（与self-attention不同，Q和V不同源，K和V同源）将conditioning融入UNet的中间层里去。

Controlnet的一个重要的功能是文生图。大框架由三个部分组成，如图。Text Encoder，负责将非结构化的，不可计算的文本内容，变成结构化可计算的向量，也就是text embedding。 Generation Mode中，将和text embedding和生成噪声组合，产生latent。 Image Decoder负责将latent还原为原始的图片，即最终的产物。通常这3个部分是分开训练，最后再组合起来的。

其中Text Encoder在controlnet中，采用的是clip text encoder。我们先来简单介绍一下clip模型。首先来看clip是如何进行预训练的，在训练的过程中，模型的输入是一个图片和文字的配对，比如图片里画的是一只狗，文本输入的是pepper，图片通过了一个图片的编码器（resNet，Vision Transformer）得到了一些特征，对于句子来说，他会通过一个文本的编码器，从而得到文本的特征。假设在每个training batch里都有n个这样的图片文本对，就会得到n个图片的特征和n个文本的特征，clip就是在这些特征上进行对比学习。对比学习需要提供正样本和负样本的定义，而在这里配对的图片文本对就是一个正样本，在这个特征矩阵里，对角线方向上的都是正样本，剩下不配对，即不在对角线上的就是负样本。

接下来就是clip如何去做零样本推理的过程。作者在这里运用了自然语言处理的方法，prompt template。clip先把标签所对应的类变成一个句子，也就是用这里的物体去替代object，产生的句子就会通过预训练好的文本编码器得到文本特征。Prompt在这里起到了文本的引导作用，在这里做prompt engineering和prompt ensembling的目的在于避免两个问题。Polysemy（多义性）问题，如果在做文本匹配的时候，只用标签所对应的一个单词进行特征的抽取，就容易碰到这样的问题，比如，crane（挖掘机，鹤）这个词，在相应的语境下，对应的含义是不一样的，这个时候就会产生歧义性。另一个问题是在预训练的时候，匹配的文本一般都是一个句子，很少出现一个单词的情况，如果推理的时候，每次进来的是一个单词，可能就存在distribution gap（即原测试集与增强后的测试集中的样本并不严格服从同一样本分布）的问题，抽出来的特征可能就不好。推理的时候，将图片输入图片编码器得到图片的特征，将图片的特征去和所有的文本特征进行相似性的计算，把图像特征和文本特征中最相近的句子挑出来，从而实现了分类的任务。Clip最为巧妙地一点在于，标签可以换成任何的单词，图片也可以选用任何图片。比如说给一张三轮车的图片，在上面的类别里给一个三轮车的类别，通过clip的零样本推理，很有可能正确的识别。但是如果像之前严格按照Image Net的1000个类去训练的分类头，模型则永远不会判断出三轮车，最多也就是把他判断成车或者自行车这些跟三轮车最接近的类。Clip彻底摆脱了categorical label的限制，也就是在预训练或者推理的时候，都不需要有提前定义的标签的列表。

Generation Mode采用Diffusion Model。Diffusion Model的运作原理是这样的，首先生成一个全是噪声的图片，接下来不断通过denoise network进行去噪，最终看到一张清晰的图片，在这个过程中，去噪的次数是实现设置好的，我们给每一次去噪一个编号，我们把这个过程称为reverse process。值得注意的是，去噪的模型是被反复使用，但是因为每次输入的图片状况差异非常大，最开始输入的就是纯粹的噪声，而在最后已经非常接近完整的图，因此还会同时输入这个编号，表示噪声的严重程度，我们希望这个模型可以根据我们输入在第几个步骤做出不同的回应。在这个模型中存在一个噪声预测器，根据输入的图片以及严重程度，生成一张噪声的图片，在用要被去噪的图片减去这个噪声，生成去噪后的结果。这么做的原因是产生一张图片和产生噪声的难度是不一样的，后者远远比前者简单的多。那么训练噪声预测器的过程也显而易见，选择一张训练的图片，随机生成噪声图，像原始图片中多次添加这张噪声图，使图片变得嘈杂。这个过程就是diffusion process。我们以一张加完噪声的图片和加入噪声的次数作为输入，加入的噪声图就是他应该预测的输出。以此为基础，在我们Text2Image过程中，在去噪过程中，我们需要同时将文本对应的embedding输入到去噪模型中，噪声预测器以及其训练的过程也同样需要输入这个内容。

Controlnet所使用的扩散模型并不是基于pixel的生成模型，而是基于latent的生成模型。他的好处在于计算效率更高，基于pixel的方法往往只能生成64\*64的图片（Imagen），再通过超分辨率模型提升图像分辨率；而基于latent能够直接生成高分辨率的图像。在Text2Image中最后的Image Decoder，我们需要先训练一个auto-encoder，输入一张图片，通过encoder将其变为latent representation，再通过decoder还原成原来的图片，输入和输出越接近越好。假设我们输入的图片为W\*H\*3，通常latent representation可以编码为h\*w\*c的，h和w是原图HW做下采样率的结果，c是通道数。训练完成后，这个decoder就能作为Image Decoder。