Attention is all your need:

传统序列模型的问题：

1. 长距离依赖：RNN等模型依赖于之前时刻的序列，所以逐步计算会丢失远距离的时间序列信息（取较大ht时，内存开销大）。
2. 不能并行：逐步计算导致不能并行处理数据。

注意力机制模型的实现：

1. 将输入序列通过嵌入层转换为固定维度的向量表示（如512）。
2. Position coding：RNN是用上一个时刻的输出作为下一个时刻的输入来作为时序信息，Transformer是使用一个正弦函数来记录下当前的位置作为时序信息，和嵌入层相加，就成功将时序信息加入到数据中。
3. 计算注意力分数：将查询（Query）、键（Key）和值（Value）向量相互作用，求向量余弦（可以有其他方式，论文里面是点乘），然后再softmax归一化得到权重。一共使用了三次注意力层，编码器一次，解码器两次。解码器的第一个注意力层使用了Mask，屏蔽掉未来时刻的位置序列，在解码器的第二个注意力层不是自注意层，K和V来自编码器的输出，Q来自解码器的第一个自注意力层的输出。这样操作可以使得解码器去在编码器的输出中挑选感兴趣的东西，也就是注意力分数大的东西。
4. MLP处理：经过Attention层后进入MLP得到输出，Attention负责把序列中的信息抓取出来做一次汇聚，MLP将他扩大到更大的语义空间时，对语义处理更加敏感。对比RNN，RNN是将上一个时刻的输出信息作为输入传递给下一个时刻，Transformer是通过一个Attention层全局的拉取序列信息，再用MLP做语义处理。

注意力机制模型的实现特点：

1. 使用注意力机制：可以避免卷积神经网络替代循环神经网络的时候出现的需要很多层卷积的问题。
2. 多头：模拟卷积神经网络多输出通道功能。
3. 自注意力机制：Q,K,V都是自身，将输入向量和所有其他位置的向量进行交互（解决长距离依赖和并行化）。
4. Mask in decoder：解码器中屏蔽掉未来时刻的位置序列（保留了传统方法中防止信息泄露的想法）。
5. Positional encoding：通过输入的时候加入时序信息，解决了注意力机制中只考虑注意力分数而忽略时序性的问题。

Deep Residual Learning for Image Recognition