Bert总结：

分成三个部分，模型的结构（中间结构和输入输出），模型的预训练方法（MLM和NSP），模型的微调方法。

模型的结构，bert和结构和transformer差不多，中间部分是12个，8头隐藏层大小为768的transformer encoder拼接起来的。每个encoder结构为：

1、self-attention（query，key，value）输入为768维，输出768维。

2、Dropout

3、Add+norm(shortcut)，输入为768,输出为768。

4、Full connected，输入为768，输出也为768。

5、Dropout

6、Add+norm(shortcut)，输入为768,输出为768。

其中norm使用的layernorm，对一个节点层计算的均值和方差做norm，例如这层节点数为batch\_size\*m,就会产生batch\_size个均值和方差。做到了无论batch\_size多大都不影响，适合训练批次比较小的情况。

输入和输出：输入是由token embedding，position embedding和segment embedding三个embedding相加组成，其中position embedding和segment embedding是通过学习得到的。输入可以是一个句子或者两个句子（用于问答任务等），两个句子的时候在两个句子的之间加入[sep]token来区分句子，而且两个句子的segment embedding不一样。输入的开头一定是一个分类[cls]token，这个token对应最后层的向量可以用于做分类，表示整个句子向量。根据对应的任务不同，最后的输出也不一样。

预训练阶段：

为了**训练深度双向表示**和**理解句子间的联系，**使用了MLM和NSP两个任务来训练模型。MLM（掩码语言模型），在输入阶段，任意选15%的单词进行预测，被选择到的单词，80%的概率替换成【mask】token，10%的概率替换成任意的token，10%的概率保持不变；在输出阶段，使用被选中token对应的输出向量做全连接+softmax预测输入的token。NSP任务，训练样本中的句子A和B，B 50%的概率是A的下一个句子，50%的概率是任意一个句子，标签就对应是和否；A，B打包在一起输入到模型，输出部分只使用【cls】token对应的输出向量做sigmoid二分类任务。

微调阶段：

微调部分只需要替换输入和输出后微调所有参数即可，根据不同的下游任务输入和输出有些不一样，对于token级别的任务输出所有token的表示，而对于分类任务，只输出【cls】对应表示加上输出层。