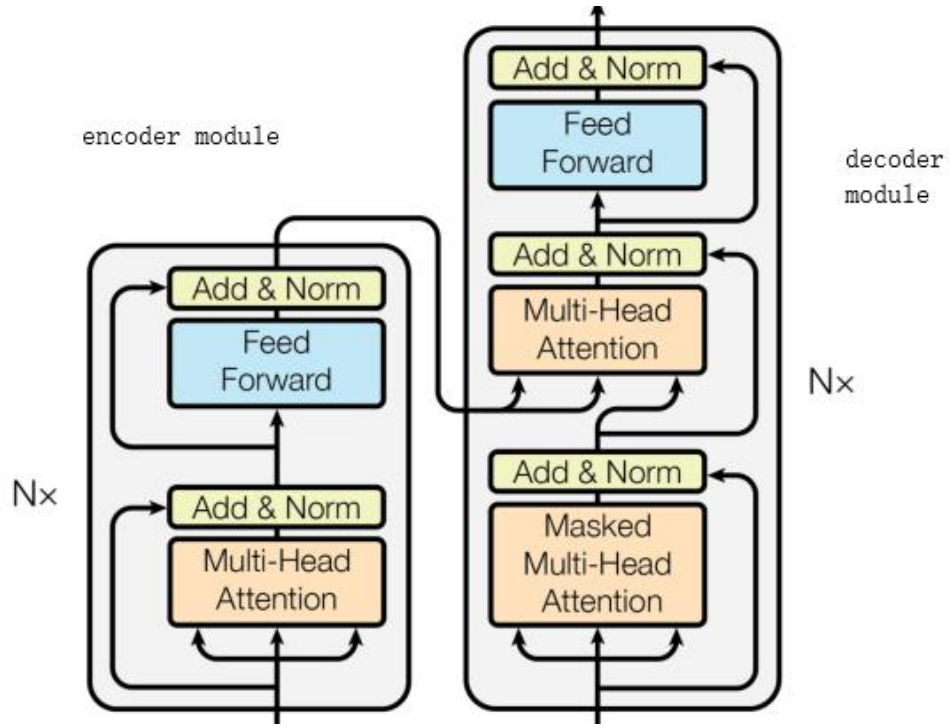
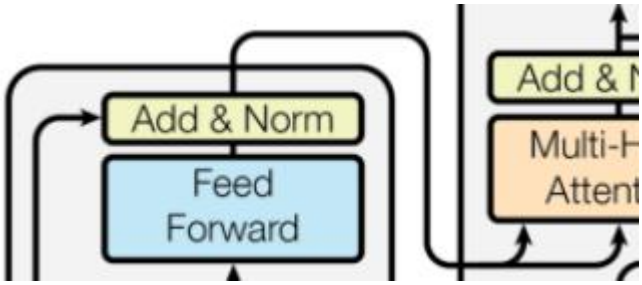
搭载 transformer 模型的基本代码思路：

一：

依据 transformer 的框架图，搭载出 Tranformer 函数：



重点理解其中 encoder 第二层传入 decoder

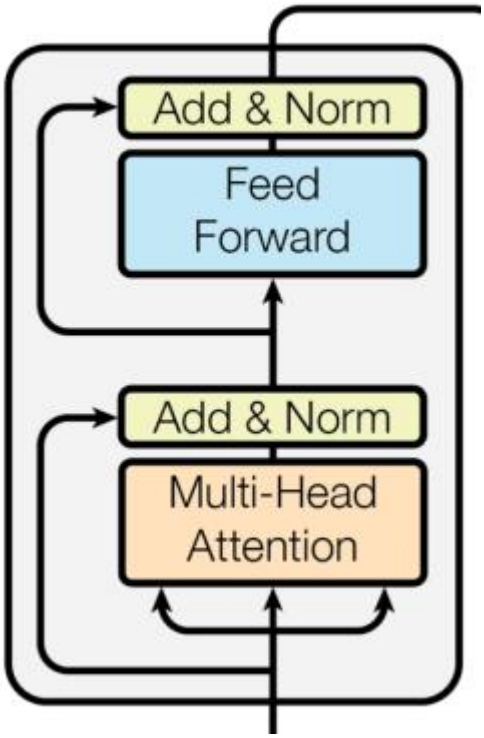



```
decode(self, memory, src_mask, tgt, tgt_mask):
```

代码中 memory 是 encoder 的运算结果传入 decoder

二：

依据框架图构件出 encoder 函数：



在 encoder module 中

Encoder 函数实现 6 词循环的神经网络大框架（该循环次数和自行定义）实现外部的框架结构，在 EncoderLayer 中具体定义了 muli-head，及 feed forward，具体实现函数是

```
#搭建框架中的 attention 层和 feedforward 层

self.sublayer = clones(SublayerConnection(size, dropout), 2)
在该函数中实现了 norm
x = self.sublayer[0](x, lambda x: self.self_attn(x, x, x, mask))

# 注意到 attn 得到的结果 x 直接作为了下一层的输入

return self.sublayer[1](x, self.feed_forward)
```

三：

定义了 sublayer module

分别实现框架图中的 add（残差连接）和 norm，及 dropout 操作。

对应代码分别是

```
return x + self.dropout(sublayer(self.norm(x)))
```

```
LayerNorm(nn.Module):
```

四：

定义 decoder 函数

Decoder 函数除了 mask 和多一层外，同时有一层是通过 encoder 传入的，具体思路同 encoder 相同。该处的 mask 是形成一个三角矩阵，在做 train 过程中，不会将 attention 注意到未求出的之后语句之中。

```
（decoder 的神经网络模块层数）self.sublayer = clones(SublayerConnection(size,
dropout), 3)
（encoder 的神经网络模块层数）self.sublayer = clones(SublayerConnection(size,
dropout), 2)
```

五：

定义 attention module：

Attention 代码中多头注意力不太好理解，具体讲解见李宏毅课程，对应的实现代码是：

```
query, key, value = \
    [l(x).view(nbatches, -1, self.h, self.d_k).transpose(1, 2)
```

这里，相当于将一个语句，拆分为多个块。分别做 attention。该处具体原因是，实测效果较好，因为能够捕获到语句中各自单词见的关系，具体讲解见李宏毅课程中，一笔带过。没有严格的数学理论支撑。

```
#此处执行多头注意力模型将数据从[14, 5, 256]变为[14, 5, 8, 32]
```

这里通过调用

```
subsequent_mask
```

来实现 mask，其中的调用较为复杂，层层嵌套。可从 run 中一步步用找函数的调用。

六：

定义 embedding 这里 transformer 中通过

```
pe[:, 0::2] = torch.sin(position * div_term)
pe[:, 1::2] = torch.cos(position * div_term)
x = x + Variable(self.pe[:, :x.size(1)], requires_grad=False)
```

增加了位置对训练的影响，实现，具体讲解内容见李宏毅课程中的讲解。

七：

Generator 函数实现将训练好的词向量映射为单词，选择概率最大的单词作为结果，使用 greed search

八：

evaluate 函数

通过调用训练好的模型，直接生成翻译结果。

参考：

李宏毅课程：

https://www.youtube.com/watch?v=ugWDIIOHtPA&t=2034s

哈佛 NLP 论坛文章及代码：

https://nlp.seas.harvard.edu/2018/04/03/attention.html

Gready search

https://blog.csdn.net/weixin_42615068/article/details/93767781 ，

该步骤每步解码后，找出最大的概率再找下一个最大的。