

学 期 2021-2022（2）

****

深度学习与自然语言处理作业

文本生成

|  |  |
| --- | --- |
| 院（系）名称 | 自动化科学与电气工程学院 |
| 学生姓名 | 黄旭聪 |
| 学号 | ZY2103203 |

2022年 6 月 17 日

1. 问题阐述

**问题：**基于Seq2seq模型来实现文本生成的模型，输入可以为一段已知的金庸小说段落，来生成新的段落并做分析。

Seq2seq是encoder-decoder结构的网络，其输入是一个序列，输出也是一个序列。其传统结构将序列中所有时刻的序列数据转换成一个固定长度的语义变量；然后对于不同时刻的输出解码器共用这个语义变量。但是这种结构，使得解码器共用一个语义变量，输入没有针对性，会造成输入的冗余和信息损失。**因此，本次大作业没有采用传统的seq2seq，而是使用了引入自注意力机制的transformer**。

1. 背景原理
   1. Transformer

由论文《Attention is All You Need》提出，现在是谷歌云TPU推荐的参考模型。它抛弃了传统的CNN和RNN，整个网络结构完全由Attention机制组成。如图1，其本质上是一个Encoder-Decoder结构，下面将递进介绍transformer的机制。

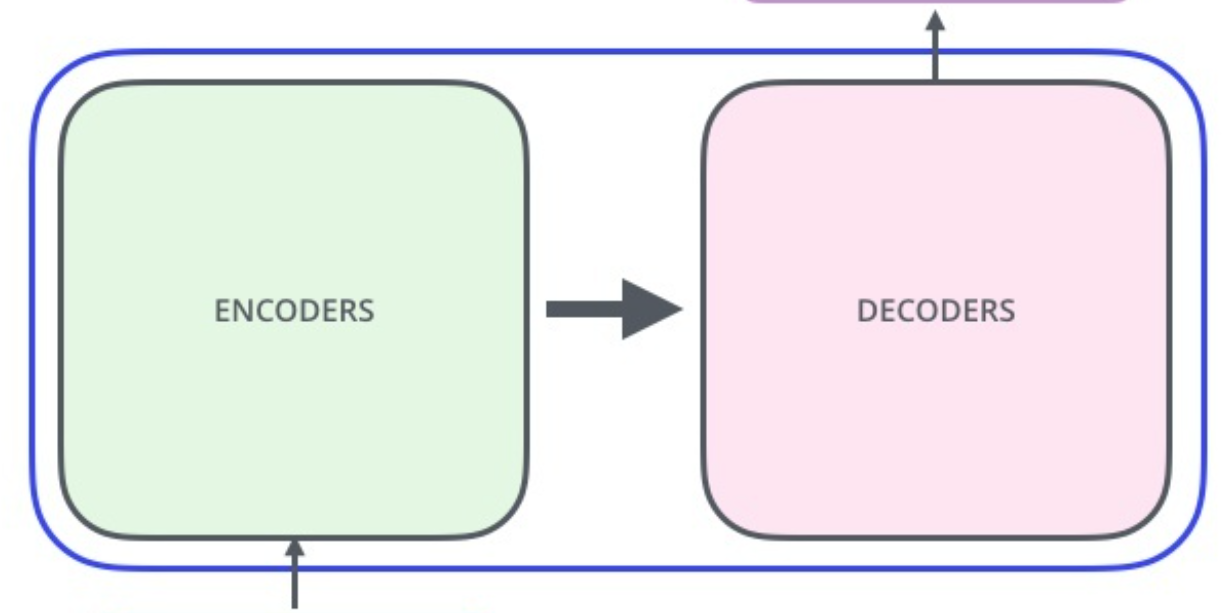


图 1 transformer整体结构

1. 整体结构

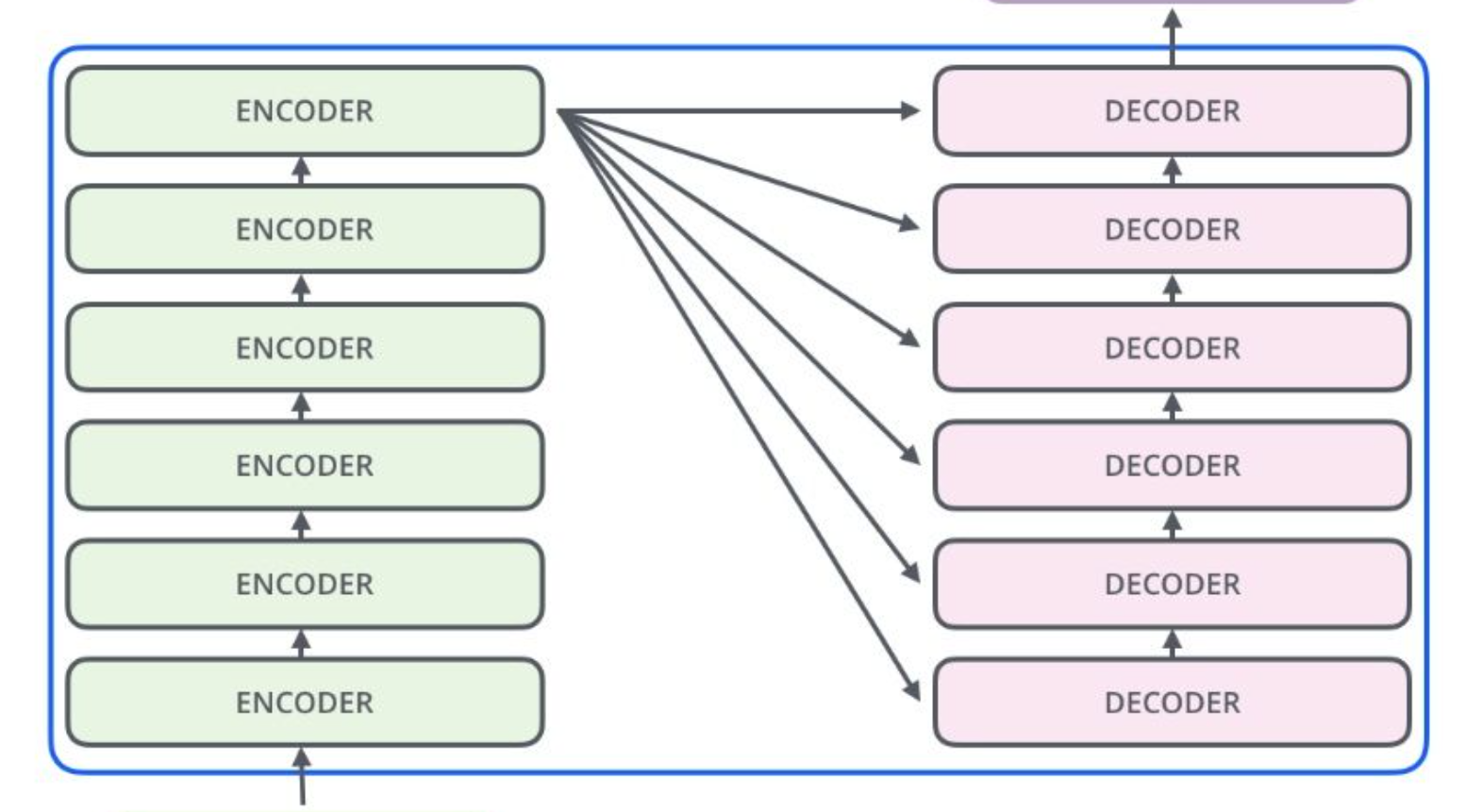


图 2 transformer堆叠结构

如图2encoder，decoder都由许多相同的块堆叠而成。每一个encoder block都可以看为一定角度下对于输入的特征提取。它们以不同的贡献作用于所有decoder block。

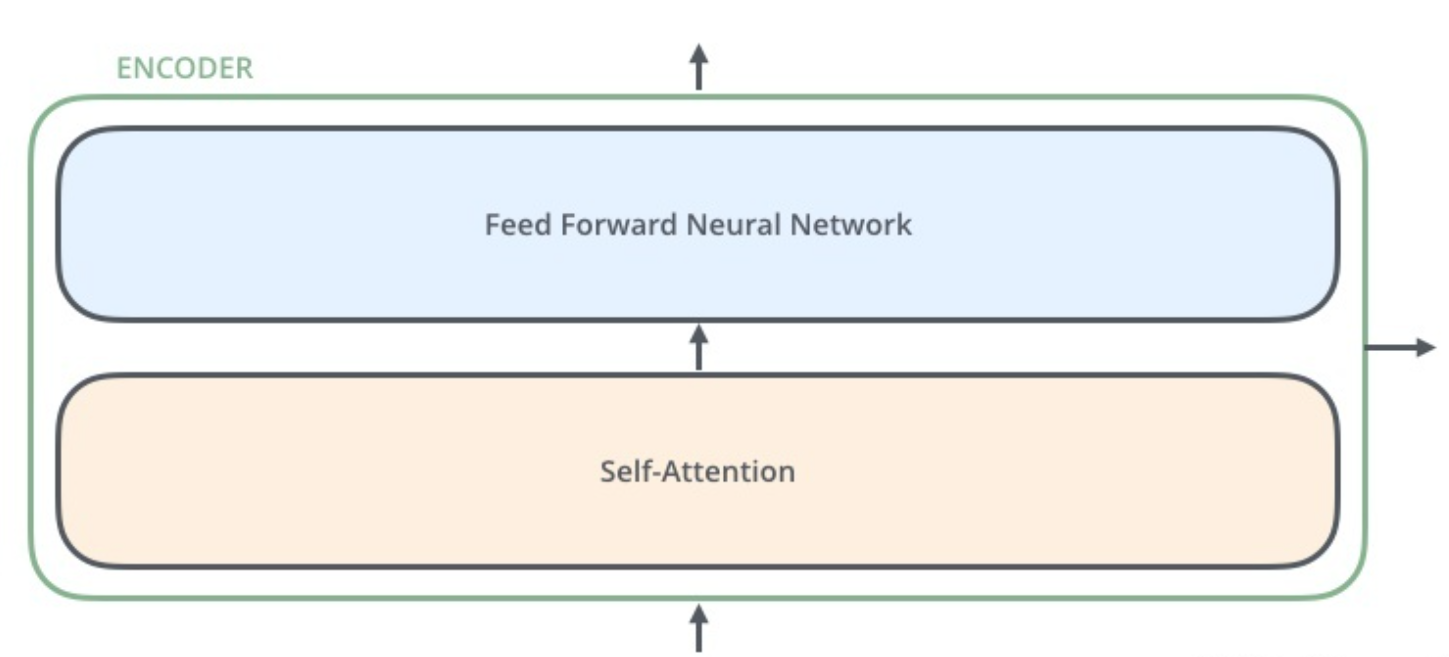


图 3 encoder构成

每一个encoder block由图3两部分组成。这种self-attention的机制的好处是两个相距较远的输入之间的距离变为了单位距离，使得具有更好的并行性。具体的，其引入了三个权重矩阵对输入进行变换，即Q,K,V阵。经下式得到当前时刻的语义向量：

语义向量经前向反馈网络送到下一个encoder block。decoder block与传统seq2seq流程相同，从词向量映射到高维词空间。

1. 关键技术

**词嵌入。**将词转化为向量，具体的参数矩阵由transformer学习得到。

**位置编码。**self-attention不能记录位置信息，所以加入位置编码，来代表不同位置的

信息，常见的编码方式有绝对编码和相对编码。相对编码不需要训练，编码之后直接加到词嵌入中；绝对编码同词嵌入需要训练。

**多头机制。**对于每个时刻的节点，设置多个生成Q,K,V的参数矩阵，每个Q,K,V学习样本的不同局部特征，最后将多个Q,K,V矩阵得到的信息拼接成语义向量。

**残差网络**。残差网络是为了使得梯度在每层网络中传播时恒大于1，从而避免梯度消失。编码器每个block中包含两个残差网络结构。分别是接在self-attention之后防止经过self-attention矩阵变换后梯度消失；接在前向反馈网络后，防止经过前向反馈网络后梯度消失。

**Attention的mask操作**。在encoder的self-attention中，由于序列长度小于最大序列长度会进行padding，故需要进行mask操作；此外，decoder中的self-attention为了避免预测时后续tokens的影响，进行mask操作；decoder中的encode-decoder attention涉及到当前token和整个encoder的计算，故encoder需要进行mask操作。

1. 实验部分

本文的训练语料为金庸的小说。具体分为数据预处理，文本转化序列，数据集生成，模型训练，相关分析。

* 1. 数据预处理

与以往的任务不同，此次需要送入网络的句子构成的batchsize。在文本清洗过程中，保留逗号，省略号等句子的标识符，剔除换行符等，句子前后加入文本开始、文本结束符，并使用占位符进行padding。详见DataPreprocess.py中的getCorpus函数。

* 1. 文本转化序列

根据文本中的字的出现频率生成字典，详见data文件夹中的basic\_vob.txt，用于词嵌入。此外，seq2seq需要序列形式的输入，故根据生成词典进一步转化文本，得到每个字的one-hot表示结果。实现one-hot表示的数字与汉字构成的字符串之间的转换。详见DataPreprocess.py中的build函数和vocabulary.py中用于文本序列间转化的类Vocab。

基于上述方法，得到每一个句子对应的序列。前一个句子为encoder的输入，后一个句子为decoder的理想输出即标签。生成的数据集对应于trainset文件夹下的train.txt和vaild.txt，最后通过Dataloader实现批次化。

* 1. 模型训练

Pytorch已提供encoder, decoder的封装，故模型总共由三部分构成：词嵌入Embedding，位置嵌入positionalEncoding以及封装好的encoder，decoder。

位置嵌入positionalEncoding使用基于正余弦函数的位置嵌入，因为sin和cos的值域是[-1,1]，可以很好地限定位置编码的大小，使得训练过程更稳定。一句话中每个单词的位置编码的同一维度上的正余弦函数的周期是相同的，不同维度的三角函数的周期是不同的。这有助于在同一维度上捕获单词之间的相对位置，而不同的维度提供了关于相对位置的不同信息（视角），从而这种编码方式能够反映句中单词的相对语序。具体如下代码：



其余部分使用pytorch封装好的词嵌入Embedding以及封装好的encoder，decoder。整体如下：



* 1. 相关分析

训练采用Intel Core i5-6200U 2.6GHZ处理器，960M GUP。模型运行60个epoch，共35000步迭代，训练集以及测试集loss变化如下：

从训练集，测试集均下降的趋势可以发现训练轮数还需增加，当前训练不充分。

下面给出输入一段话后，产生的对应输出：

输入：段誉见这两人都是肩悬药囊，手执一柄刃身极阔的短刀

生成段落：段誉道脚下也甚迅捷，只盼能抓到什么东西，这么乱挥一阵。倘若一幌即回，石底发出藤萝声音，知道大小岩石缠结。住得有人，那可奇怪之极了。

1. 总结

本学期的课程到此为止了，但是秦老师传授的知识将使我始终收益。无论是线性代数的创新型认知，EM算法的数学理论推导，EMM算法的讲解再到语言模型，深度学习。这些知识不再是生硬的文字，变成了随之理解的记忆。对于我今后运用深度学习，进行数学推导等大为受用。

这门课理论与实践相结合，不时还穿插着秦老师的观点，见解。最令我收益的秦老师所说的：“没有对错，只有好坏，只要能解释得通就行”。科研应该如此，不是继承，而是吸收并创新。

最后，感谢秦老师的辛苦付出，也祝愿这门课程越来越好！！

1. 附：

Github链接：https://github.com/189223/NLP\_Assignment4