1. BERT是如何使用Transformer的?

BERT 只使用了 Transformer 的 Encoder 模块，原论文中，作者分别用 12 层和 24 层 Transformer Encoder 组装了两套 BERT 模型：

$$BERT_{base}：L=12, H=768, Head=12, Parameters=110M$$

$$BERT_{large}：L=24, H=1024, Head=16, Parameters=340M$$

同时，BERT的Embedding也进行了相应调整，多了 Segment Embeddings，同时position embedding也是可训练的。

2. BERT的训练任务

- MLM

  一句话中随机选择 15% 的词汇用于预测。对于在原句中被抹去的词汇， 80% 情况下采用一个特殊符号 [MASK] 替换， 10% 情况下采用一个任意词替换，剩余 10% 情况下保持原词汇不变（不单单是根据上下文预测该单词，而且赋予了模型一定的纠错能力, 同时能缓解与fine-tune阶段不一致问题）。

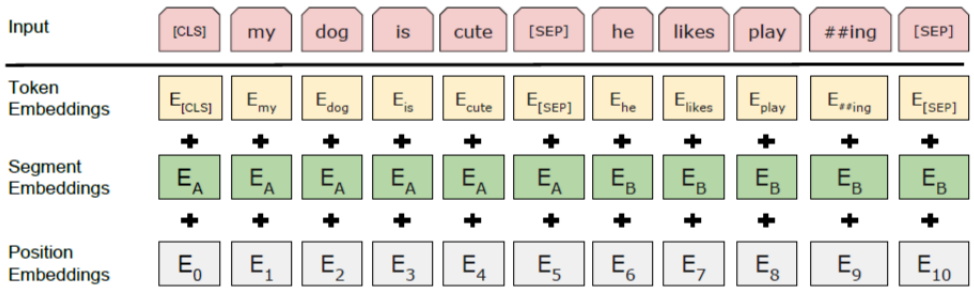
- NSP

  判断两句话是否是前后句，正样本是顺序的两句话，副样本是随机的两句话。模型学到的更多的是topic方面的知识，对预训练语言模型帮助不是很明显，后续的其他预训练模型大多也都去除或者替换的该任务。

3. BERT有什么局限性

- 因为采用wordpiece分词，所以[MASK]掉的可能并不是完整的词，这样的预测可能会较为容易一点，对此也有相应的whole word mask方法
- [MASK]掉的词没有考虑固定词组情况，比如：New York，对此也有提出span mask的方法 *（spanBERT， 随机mask片段）*
- BERT的在预训练时会出现特殊的[MASK]，但是它在下游的fine-tune中不会出现，这就出现了预训练阶段和fine-tune阶段不一致的问题。 *（MASK的百分比策略能够一定程度缓解这个问题）*
- NSP任务的局限性(ALBERT SOP)

4. BERT的Embedding（输入形式）



BERT的embedding由三部分**随机生成可训练的**的embeding构成构成：

- Token embedding：词本身的embedding表示
- Postition embedding：表示当前词对应的位置，从0～512
- Segment embedding： 表示当前词属于哪一句话，从0开始

BERT中的position embedding不同于Transformer中的position encoding，其不是根据词位置由sin/cos直接计算得出，而是根据其词位置初始化的d维可训练向量。作者这里这样选择的原因可能是 BERT 训练的数据比 Transformer 那篇大很多，完全可以让模型自己去学习位置关系。

5. BERT的三种embedding相加，会对语义产生影响吗?

没有影响

坦白讲这个问题我也没有完全想通，我觉得唯一合理的解释是空间维度很高，所以模型总能分开各个组分。

6. BERT能否有效处理空格丢失的文本

应该不能，可以先做分词处理

7. BERT应用于单词拼写错误的数据是否有效

错误量小的情况下，影响应该不大，因为预训练的海量预料中极有可能包含这些错误。如果错误量过大，可能需要先纠正文本

8. BERT为什么要MASK？具体是如何MASK的?

BERT 通过在输入 X 中随机 Mask 掉一部分单词，然后预训练过程的主要任务之一是根据上下文单词来预测这些被 Mask 掉的单词。其实这个就是典型的 DAE(Denosing Autoencoder) 的思路，那些被 Mask 掉的单词就是**在输入侧加入的所谓噪音**。

一句话中随机选择 15% 的词汇用于预测。对于在原句中被抹去的词汇， 80% 情况下采用一个特殊符号 [MASK] 替换， 10% 情况下采用一个任意词替换，剩余 10% 情况下保持原词汇不变（不单单是根据上下文预测该单词，而且赋予了模型一定的纠错能力, 同时能缓解与fine-tune阶段不一致问题）。

9. MASK和CBOW的异同点

> 什么是cbow?
>
> cbow是一种训练词向量的方式，word2vec通常用cbow和skip-gram两种实现方式：
>
> cbow是利用周围词预测中心词，skip-gram是利用中心词来预测周围词

相同点：

- 都是根据上下文去预测当前词

不同点：

- CBOW是去掉当前词，用上下文去预测当前词，一句话中的每个词都会成为当前词。而BERT是一次性mask 15%的词，这样在大规模数据下训练周期会短一些。
- CBOW得到的每个单词的embedding是唯一的，并不能很好的处理一词多义问题。而BERT得到的embedding融合了上下文信息，在不同语境下，word embdding是不同的。

10. BERT两个训练任务的损失函数是什么?

> 负对数似然损失（负对数似然损失和交叉熵损失从最终公式看是相同的，两者殊途同归，但是出发点不通，这里我没查到很好的具体资料解释二者异同。pytorch代码实现上用的就是交叉熵损失函数）
>
> 注：θ：BERT 中 Encoder 部分的参数；θ1：是 Mask-LM 任务中在 Encoder 上所接的输出层中的参数；θ2：是句子预测任务中在 Encoder 接上的分类器参数；

- MLM: $L(\theta,\theta_1)=-\sum_{i=1}^{M}log(p(m=m_i|\theta,\theta_1)),m_i\in[1,2,3...|V|]$
- NSP: $L(\theta,\theta_2)=-\sum_{j=1}^{N}log(p(n=n_j|\theta,\theta_2)),n_j\in[isNext,notNext]$

$$L(\theta,\theta_1,\theta_2)=-\sum_{i=1}^{M}log(p(m=m_i|\theta,\theta_1))-\sum_{j=1}^{N}log(p(n=n_j|\theta,\theta_2))$$

11. BERT中的激活函数

BERT中使用Gelu激活函数 (Transformer使用的是Relu)，$Gelu(x)=x\phi(x)$,其中，ϕ是正态分布

Gelu激活函数引入了随机正则的思想（类似于dropout），$\phi(x)$决定x中有多少信息被保留，$\phi(x)$越大，x越有可能保留，反之，越有可能置0。其在nlp模型中，表现较好。

12. BERT中为什么要使用 [CLS]?

BERT 在第一句前会加一个 [CLS] 标志，最后一层该位对应向量可以作为整句话的语义表示，从而用于下游的分类任务等。这个无明显语义信息的符号会**更“公平”地融合文本中各个词的语义信息**，从而更好的表示整句话的语义。

补充：BERT源码中

- get_pooled_out 即得到[CLS]的表示，shape= [batch, hidden size]
- get_sequence_out得到所有词的表示，shape = [batch, sequence length, hidden size]

13. 词带模型到word2vec改进了什么？word2vec到BERT又改进了什么?

词袋模型表示方式不考虑**文法以及词的顺序**。在用词袋模型时，文档的向量表示直接将各词的词频向量表示加和。

word2vec 其底层主要采用基于 CBOW 和 Skip-Gram 算法的神经网络模型，word2vec考虑了词与词之间的顺序，引入了上下文的信息，将每一个词语映射成一个低维稠密向量。

BERT利用更深的模型，以及海量的语料，得到更为准确的embedding表示，同时，在不同的上下文环境下，得到的word embedding是不一样的

14. BERT为什么比ELMO效果好？BERT和ELMO的区别

为什么BERT效果好：

- LSTM的抽取能力远弱于Transformer
- BERT的训练数据与参数规模远大于ELMO

区别：

- ELMO是通过语言模型得到训练数据中句子中所有单词embedding表示concat到下游任务的embeeding中，以此做为下游任务的特征去使用。（**feature-base pretrain**）
- 而BERT是基于“fine-tune”模式的预训练语言模型，下游任务需要使用BERT模型及参数。**(fine-tuning pretrain)**

15. 使用BERT预训练模型为什么最多只能输入512个词，最多只能两个句子合成一句?

这是Google BERT预训练模型初始设置的原因，前者对应Position Embeddings，后者对应Segment Embeddings。

也就是说，google在预训练该模型时超参就是这样设置的。

16. BERT分词做了哪些事?

- Basic 分词： 根据空格划分英文单词等一些预处理
- Wordpiece分词：切分为细粒度的词根词缀等

17. 如何解决文本溢出问题（长度超过512）

- 首位截取：只保留前510长度（留两位给cls和sep）
- 尾部截取：只保留后510长度
- 首尾截取：保留头部n，和尾部m
- 滑动窗口：把文档分成**有重叠**的若干段，然后每一段都当作独立的文档送入BERT进行处理。最后再对于这些独立文档得到的结果进行整合（可以用cls融合等方式）
- 等等...

18. BERT非线性的来源在哪里？multi head attention 是线性的嘛?

FFN的gelu激活函数和self-attention，self-attention是非线性的

19. 如何优化BERT?

结合RoBERTa、ALBERT、ERNIE等说明