一种强大的预训练模型：BERT

指导老师：幹泰彬　　周方全 | 202121081229　　研究方向：自然语言处理

摘要

BERT(Bidirectional Encoder Representation from Transformers) 是基于深层 Transformer 的预训练模型。在自然语言呀就领域中，它是一种非常强大的模型，近年来绝大多数自然语言处理的模型都是在其的基础上进行改进优化的。BERT 不仅充分利用了大规模无标注文本来挖掘其中丰富的语义信息，同时还进一步加深了自然语言处理力模型的深度。本文着重介绍在 BERT 模型中主要成分 Transformer 模块的数学推导过程以及 BERT 建模的方法。最后简要介绍使用 BERT 进行文本分类。

关键词- BERT; Transformer; 自然语言处理; 文本分类;

1 介绍

在 BERT 出现之前，无论是基于卷积神经网络还是循环神经网络的语言模型在面对自然语言处理的问题时都是一个浅层的神经网络语言模型。然而，BERT 作为一个种非常具有深层思维结构的预测和训练程式语言编程的模型打破了这个壁垒。其“里程碑”的巨大重要性和历史意义就体现在于：不仅证明了针对自然语言处理训练中的分析精度和预测准确率也同样可以通过多次增加数据模型的预测深度而显著地得到提高，模型也同样可以通过对一些无符号标记的海量数据模型进行多次集中预测和训练从而获得，使得模型训练方便。

根据论文 [1] 的介绍，在相同的任务中 BERT 模型相比其它模型在准确率上会有明显的提高。BERT 的通过巧妙的方式获得文本的上下文信息，从而帮助计算机去理解那些在文章中模棱两可的语言。有时在一段话中我们知道有些词句是这段话的重点，它代表了这段话的主要含义；有时相同的一句话在不同的上下文环境中会有完全不同的意思；有时一句话中代词的理解不过也会是文章的意思发生巧妙的变化。这些在 BERT 模型中都有了很大程度的解决。下面我们先来介绍一下 BERT 的整体结构。

2 整体构架

BERT 的基本模型结构由多层 Transformer 构成，包含两个与训练任务：掩码语言模型（Masked Language Model, MLM）和下一个句子预测（Next Sentence Prediction, NSP），如图 1所示。

可以看到，模型的输入由两段文本 $x^{(1)}$ 和 $x^{(2)}$ 拼接组成，然后通过 BERT 建模得到上下文语义表示，最终学习掩码语言模型和下一个句子预测。需要注意的是，掩码语言模型对输入形式并没有特别的要求，可以是一段文本也可以是两段文本。而下一步句子预测要求模型的输入是两段文本。因此，BERT 在预测训练阶段的输入形式统一为两段文本拼接的形式。接下来我们介绍如何对两段文本建模，得到对应的输入表示。

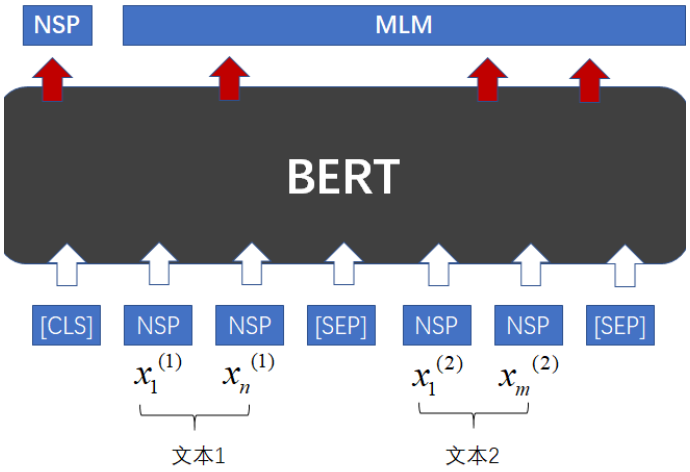

图 1: BERT 的整体模型结构

BERT 的输入表示由 Token，Segment，和 Position 三部分构成，因为他们要相加，所以单个向量的维度必须是相等的，这样才能相加，如图 2所示。

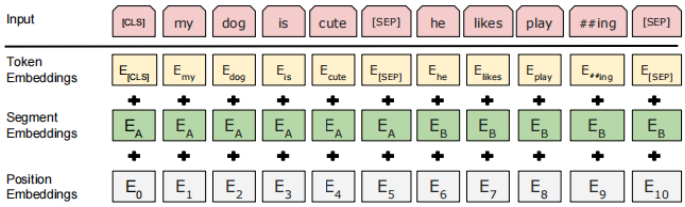

图 2: BERT 的输入表示

Token 中包括 CLS 和 SEP。CLS 就是 classification，模型最后一层输出的 CLS 提取出来，可以拿他做分类。SEP 就是 separator，用来分隔两段话，放在每段话结尾。因为 NLP 中有的任务是单句输入，有的是多句输入，所以 BERT 为了处理各种任务，设置了 SEP 进行分割，用一套框架就能处理所有任务。

Segment embedding 用来区分每个 token 属于前一段话还是后一段话，是一个可学习的嵌入向量，与 SEP 一起，用来区分不同句子。

BERT 的 Position embedding 与 Transformer 的不同，Transformer 的是由正余弦计算出来的，可以外推，而 BERT 的是个可学习的嵌入向量，学好后维度不能变，没法外推，所以 BERT 会有输入长度限制。

3 Transformer 模块的数学推导

3.1 基本模块介绍

在论文 [2] 中提出了 Transformer 模型，下面我们具体介绍一下 Transformer。首先假设 Transformer 模型就是一个黑箱，我们以机械翻译这个应用为例看看 Transformer 模型是怎么样工作。如图 3，喂入一个输入，给出一个相应的输出。

我们把这个黑箱展开就会发现里面有两个重要的组件：Encoders 和 Decoders，以及连接这两个组件的连线。而在 Encoders 组件中有若干个一样结构的 encoder 层，同样 Decoders 组件中也有若干个一样结构的 decoder 层，如图 4所示。

如图 5，每一个 encoder 层又可以被细分为两个子层：self-attention 层和前馈神经网络层。

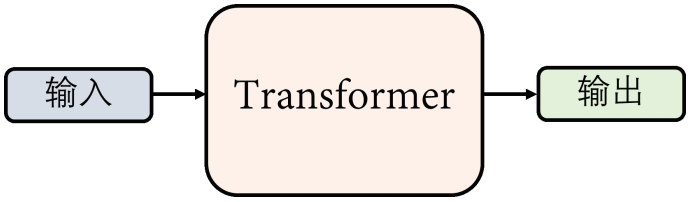

图 3: Transformer 模型

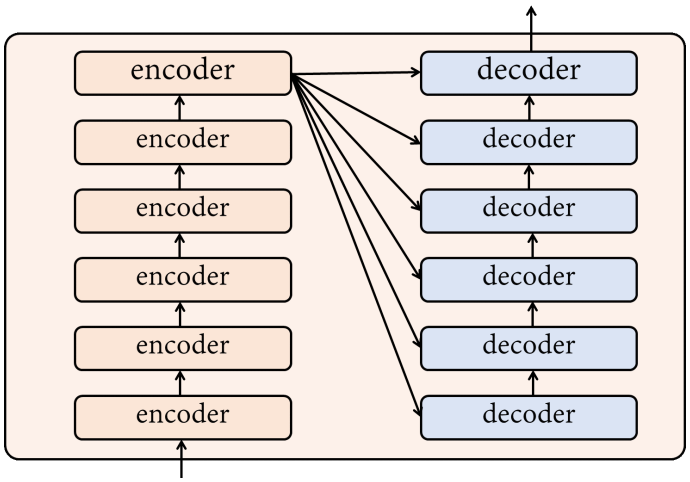

图 4: 在图中显示的是 6 个 encoder 层和 decoder 层

输入首先经过的是 self-attention 层，该层帮助编码器在编码特定单词时查看输入句子中的其他单词。该层在后面会进一步讲解。self-attention 层输出的数据喂入后面的前馈神经网络层；每个 encoder 层可以被分成三个子层：self-attention 层、attention 层和前馈神经网络层。其中第一层和第三层是和 encoder 层是一样的，中间多加了一个 attention 层。这个增加的层会使 decoder 层的注意力分到输入的相关序列上去。

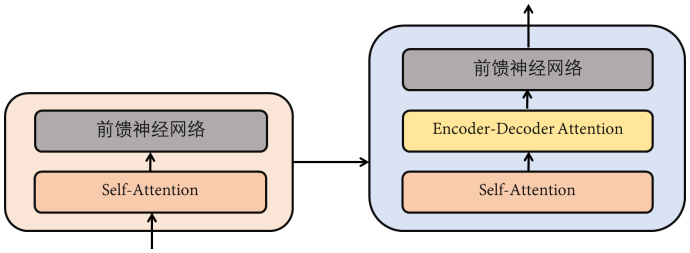

图 5: 左边是 encoder 层，右边是 decoder 层

3.2 主要模块 Self-Attention 中的数学推导

接下来我们来介绍一下在 encoder 层和 decoder 层都有的子层 Self-Attention 层。假设我们对该模型输入一个句子，该句子有 d 个单词，表示为 $S = (s_1, ..., s_i, ..., s_d)^T$ 。其中 S 表示一个句子，s_i 表示单词。

如图 6所示。首先，我们使用 CWOB 将句子 S 进行编码得到改句子的向量形式 X。其中 $\mathbf{X} = (x_1, ..., x_i, ..., x_d)^T$，$x_i$ 表示对每个单词的编码，他们的维度都是一样的。这个维度要看 CBOW 的设置值，本文设置为常数 n，故 $\mathbf{X} \in \Re^{d \times n}$ 。而且这个词编码过程只在 Encoders 组件中第一个 encoder 层中的 Self-Attention 层的输入才进行，之后的 encoder 层的输入不再进行 CBOW 编码。X 经过 Self-Attention 层后得到输出为 $\mathbf{Z} = (z_1, ..., z_i, ..., z_d)^T$ 。之后 Z 会进入

前馈神经网络层，他的输出会成为写一个 encoder 层的输入。

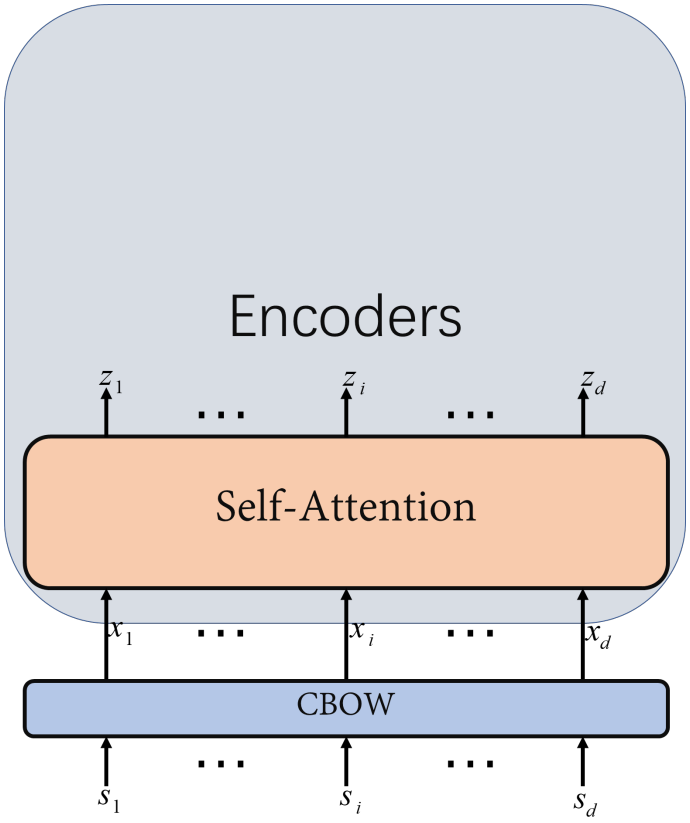

图 6: 句子 S 经过 CBOW 编码得到 X，X 经过 Self-Attention 层后得到 Z 层

由图 6我们可以直观的看出每一个词在经过 Self-Attention 层时都有其自己的路径。而且在 Self-Attention 层中这些路径存在一些依赖关系。而在前馈神经网络层不存在这样的路径依赖关系，因此不同路径在前馈神经网络层的并行计算成为了可能。接下来我们看看在子层 Self-Attention 中到底是怎么运作的。由于每个词向量都有自己的路径，所以我们以词向量 $x_i \in \Re^{1\times n}$ 为例看看在 Self-Attention 层是如何计算的。如图 7可以帮助我们接下来四步的计算过程。

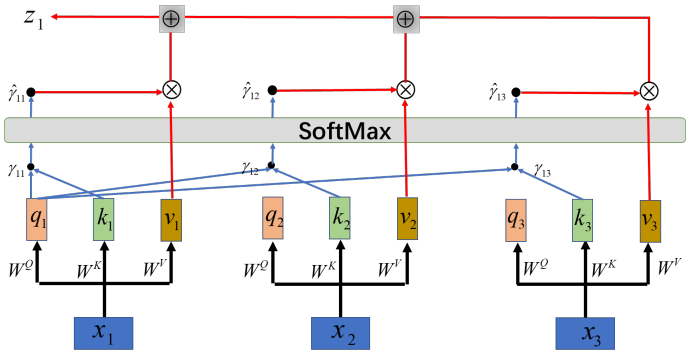

图 7: 这个句子有 3 个词，以第一个词向量为例看看其在 Self-Attention 子层中经过计算得到第一个输出的过程

i. 首先我们设置三个待训练的参数矩阵：Query 矩阵 $\mathbf{W}^Q{}_{n\times m}$，Key 矩阵 $\mathbf{W}^K{}_{n\times m}$ 和 Value 矩阵 $\mathbf{W}^V{}_{n\times m}$。他们都是 m 列的参数矩阵，且所有的词向量公用这三个参数矩阵，一般而言 $m \leq n$。接着我们用词向量分别和三个参数矩阵做矩阵相乘得出三个向量：Query 向量 q_i，Key 向量 k_i 和 Value 向量 v_i，其中 q_i 、k_i 、$v_i \in \Re^{1\times m}$，。这三个向量是计算和

思考注意力的抽象概念。其中 $q_i = x_i \bullet \mathbf{W}^Q$，$k_i = x_i \bullet \mathbf{W}^K$ ，$v_i = x_i \bullet \mathbf{W}^V$ 。同样，你可以使用相同的方法得到其他词向量的 q、k 和 v。

ii. 我们需要句子中每个单词对该单词进行一个评分，这个评分决定我们在处理这个词向量时，注意力在句子其他部分的关注程度。我们通过公式(1)计算出我们在处理词向量 x_i 时，各单词的评分 $\alpha_i = (\gamma_{i1}, ..., \gamma_{ij}, ..., \gamma_{id})$。其中，$d_k$ 是向量 k_i 的维度。

$$\alpha_i = q_i \bullet \left({k_1}^T, ..., {k_j}^T, ..., {k_d}^T\right) \Big/ \sqrt{d_k} \tag{1}$$

iii. 接着我们将得到的 $\alpha_i \in \Re^{1\times d}$ 进行 SoftMax 处理。即 $\hat{\alpha}_i = \text{SoftMax}(\alpha_i)$，使得 $\sum_{j=1}^{d} \hat{\gamma}_{ij} = 1$ 。此时就能很清楚地看出哪个词向量的注意力分配的更多，注意这里的 SoftMax 是以行为单位进行计算的，这在后面矩阵处理中会用到。

iv. 最后我们将得到的权重 $\hat{\alpha}_i$ 分别和 v_i 进行数乘然后再加得到最后的输出 z_i ，其计算如公式(2)所示，可知 $z_i \in \Re^{1\times m}$ ，其中 $v_j \in \Re^{1\times m}$，$\hat{\gamma}_{ij}$ 是一个 0 1 的常数。这样做会使相关度高的词保留下来，而相关度低的词由于权重太小而被忽略。

$$z_i = \sum_{j=1}^{d} v_j \times \hat{\gamma}_{ij} \tag{2}$$

通过上面这四步，我们就得到了词向量 x_i 经过 Self-Attention 子层只有得到的输出 z_i，继而将得到的结果输入后面的前馈神经网络子层。而在实际的操作中我们并不是一个词向量一个词向量来运算的，而是将一个句子的词向量组成一个矩阵一并送入 Self-Attention 子层来进行计算。这样就可以实现不同词向量的并行化。针对矩阵的运算我们也有相应的硬件 GPU 来对矩阵运算进行加速。下面我们看怎么在矩阵层面来进行 Self-Attention 子层里面的运算。

我们输入一个句子 $\mathbf{X} = (x_1, ..., x_i, ..., x_d)^T \in \Re^{d\times n}$ 矩阵，最后得到 $\mathbf{Z} = (z_1, ..., z_i, ..., z_d)^T \in \Re^{d\times m}$。首先我们定义 Query 向量组 q_i，Key 向量组 k_i 和 Value 向量组 v_i 组成的矩阵分别是 $\mathbf{Q}_{d\times m}\mathbf{K}_{d\times m}$ 和 $\mathbf{V}_{d\times m}$。它们的计算公式如下：

$$\mathbf{Q} = \mathbf{X} \bullet \mathbf{W}^Q \tag{3}$$

$$\mathbf{K} = \mathbf{X} \bullet \mathbf{W}^K \tag{4}$$

$$\mathbf{V} = \mathbf{X} \bullet \mathbf{W}^V \tag{5}$$

然后通过如下公共得出 $\mathbf{Z} = (z_1, ..., z_i, ..., z_d)^T \in \Re^{d\times m}$：

$$\mathbf{Z} = SoftMax\left(\frac{\mathbf{Q} \bullet \mathbf{K}^T}{\sqrt{d_k}}\right) \bullet \mathbf{V} \tag{6}$$

其中，SoftMax 函数对括号里面的矩阵是以行为单位进行计算的。

4 BERT 进行文本分类

文本分类是多个文本处理应用程序的基础，并被用于许多不同的领域，如市场人力资源、CRM、研究和科学 (患者医疗状态分类)、社会网络监控 (实时紧急监测、虚假信息发现或任何

冒犯性的评论等)。由于深度学习 NLP 技术的出现，文本分类模型取得了显著的效果，其中 BERT 模型和其他模型发挥了主导作用。

我们一定想要用 BERT 开辟的各种可能性。我们有很多方法可以利用 BERT 的大型知识库来实现我们的 NLP 应用。BERT 是一个可以微调的模型，我们可以根据自己的任务和特定于任务的数据对模型进行微调，使模型在这样一组特定的参数下能满足我们想要的应用。那么面对一系列文本我们怎样处理呢？

（1）我们需要把得到的原始文本进行清洗——预处理，将句子中的符号，标点，无实意的词去掉。（2）我们将清洗好的文档结构化。（3）我们将文本输入 BERT 模型得到相应的词向量编码，由模型得到的高维度的向量表达能力更强，更能显示词与词之间的关系以及次在文本中的关系，所以同样的词但不同的文本环境中得到的词向量编码是不一样的数值。（4）将得到的词向量喂入分类的神经网络，经过训练在得到一个模型参数。这个模型就可以对相应的应用范围批量的对大量文本进行分类。

此外，如果我们的原始数据具有一定的感情色彩标签，那么我们经过训练就可以得到一个可以分析情感倾向的模型。这在餐饮评价，电影评价以及电商的商品评价有很大的应用价值。

5 展望

今年来基于词向量语言模型的自然语言处理任务大多离不开 BERT，因为它真的很强大。我们对这个模型的依赖在于，它像是一个很好用基础的工具，在此基础上我们可以把原来基于词向量的任务进行优化和提高。再加上近年来深度学习，神经网络风靡一时，以词向量为基础的自然语言任务越来越多，这与能生成高质量的词向量模型 BERT 相得益彰。除非再出现一个超越 BERT 的模型，不然它仍会在 NLP 领域中发挥着无可代替的作用。

参考文献

[1] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[2] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.